

**DORMAN Computer Program
(Study 2.5) Final Report
Volume IV
DORMAN Program Listing, UNIVAC 1108 Version**

Prepared by DATA PROCESSING SUBDIVISION
Information Processing Division

11 September 1973

(NASA-CR-137358) DORMAN COMPUTER PROGRAM
(STUDY 2.5). VOLUME 4: DORMAN PROGRAM
LISTING, UNIVAC 1108 VERSION Final
Report (Aerospace Corp., El Segundo,
Calif.) 97 p HC \$8.00

CSCL 09B G3/08

67302122324252627282930311234567890
APR 1974
N74-19830

Unclas
15928

Prepared for OFFICE OF MANNED SPACE FLIGHT
NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
Washington, D. C.

Contract No. NASW-2472



Engineering Science Operations
THE AEROSPACE CORPORATION

Aerospace Report No.
ATR-74(7335)-1, Vol. IV

DORMAN COMPUTER PROGRAM (STUDY 2.5) FINAL REPORT
Volume IV. DORMAN Program Listing, UNIVAC 1108 Version

Prepared by
Data Processing Subdivision
Information Processing Division

11 September 1973

Engineering Science Operations
THE AEROSPACE CORPORATION
El Segundo, California

Prepared for
OFFICE OF MANNED SPACE FLIGHT
NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
Washington, D. C.

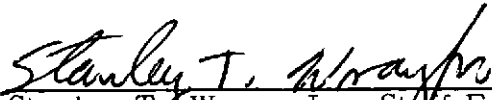
Contract No. NASW-2472


;

DORMAN COMPUTER PROGRAM (STUDY 2.5) FINAL REPORT


Volume IV: DORMAN Program Listing, UNIVAC 1108 Version

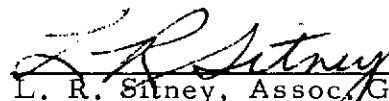
Prepared by


Stanley T. Wray, Jr., Staff Engineer
Advanced Application Staff
Data Processing Subdivision


L. Sashkin, Director
Data Processing Subdivision
Information Processing Division

Approved by


Robert R. Wolfe
NASA Study Director
Advanced Vehicle Systems
Directorate
Systems Planning Division


L. R. Silney, Assoc. Group Director
Advanced Vehicle Systems
Directorate
Systems Planning Division

PRECEDING PAGE BLANK NOT FILMED

FOREWORD

Study 2.5, DORCA Applications, has been directed at development of a data bank management computer program identified as DORMAN. The size of the DORCA data files and the manipulations required on that data to support analyses with the DORCA program necessitates automated data techniques to replace time consuming manual input generation. The DORCA program (Dynamic Operations Requirements and Cost Analysis) was developed by The Aerospace Corporation for use by NASA in planning future space programs. Both programs are designed for implementation on the UNIVAC 1108 computing system at the NASA Computing Facility, Slidell, Louisiana.

This volume contains a listing of the UNIVAC 1108 version of the DORMAN program. The code printed herein has been compiled, loaded, and executed successfully on the EXEC 8 system for the UNIVAC 1108 at Slidell, Louisiana. This was accomplished during the month of September (1973) using the segment map contained in Volume II of this report.

In addition to this volume, the following documentation has been prepared.

Volume I	Executive Summary
Volume II	User's Guide and Programmer's Guide
Volume III	Original Data Bank Listing

Study 2.5, DORCA Applications, is one of several study tasks conducted under NASA Contract NASW-2472 in FY 1973. The NASA Study Director was Mr. V. N. Huff, NASA Headquarters, Code MTE.

By agreement with Mr. Huff, the DORMAN program will be delivered directly to the NASA Computing Facility.

*ELT,I DORMAN,BLOCK	BLOCK 3
BLOCK DATA	BLOCK 5
COMMON /BFRS/ ISYN(2,20),ISYN1,IT1,IT2,IT3,FULL,CN1,CN2,LIN1,LIN2	BFRS1 2
* ,NWBUF,NB1,NB2,ICN1,ICN2	BFRS1 3
COMMON /BFRS/ BUF1(14,50),BUF2(14,50),AAA(700)	BFRS1 4
INTEGER CN1,CN2,BUF1,BUF2	BFRS1 5
LOGICAL FULL	BFRS1 6
COMMON /MISC/ERFLAG,FERR,KARD(14),ACTION(14)	MISC 2
INTEGER ERFLAG,FERR,ACTION,PRTFIL	MISC 3
EQUIVALENCE (PRTFIL,FERR)	MISC 4
COMMON /NAMES/ IVER,DNAME(2),MODNAM(2),BNAME(2)	NAMES 2
INTEGER DNAME,MODNAM,BNAME	NAMES 3
COMMON /FILES/ BASIC,MTAPE,FINAL,S1,S2	FILES 2
INTEGER BASIC,MTAPE,FINAL,S1,S2	FILES 3
COMMON /REST/TABLE,USES,FILE,END,DECK,DDECK,BLANK ,BATCH	REST 2
INTEGER TABLE,USES,FILE,END,DECK,DDECK,BLANK,BATCH	REST 3
COMMON/VCARD/ILBL(14)	BLOCK 11
COMMON /WORK/IIUNT(3),IIRW(3),IUTBL(3,19),IACT(8,3),NFILES	WORK 2
DATA (ILBL(I),I=1,14) /	BLOCK 13
* 6H\$DORMA, 6HN DATA, 6H BANK , 6H , 6HVERSIO, 6HN ,	BLOCK 14
* 1H ,1H ,1H ,1H ,1H ,1H ,1H ,1H /	BLOCK 15
DATA FINAL,BASIC/14,1/,PRTFIL/6 /	BLOCK 16
DATA USES, BLANK, END / 6H USES , 6H , 6H\$END 0 /	BLOCK 17
DATA TABLE/6HF TABL/,DECK/6HF DECK/,DDECK/6H\$DECK /,FILE/6HF FILE/BLOCK	BLOCK 18
DATA BATCH/0/	BLOCK 19
DATA IUTBL/	BLOCK 20
* 0,0,0, 1,2,0, 2,2,2, 3,2,2,	BLOCK 21
* 4,2,1, 5,3,0, 6,2,1,	BLOCK 22
*11,2,2, 12,2,2, 13,2,2, 14,2,2,	BLOCK 23
*21,2,2, 22,2,2, 23,2,2, 24,2,2	BLOCK 24
*25,2,2, 26,2,2, 27,2,2 , 28,2,2	BLOCK 25
* /	BLOCK 26
DATA NFILES/19/	BLOCK 27
DATA IVER /4HTEMP/	BLOCK 28
DATA IACT/24*0/	GWT25 2
END	BLOCK 29

#ELT,I DORMAN.DORMAN

DIMENSION IU(3),IR(3)

INTEGER CONTEN,GENEOL

INTEGER CREATE,USE,OPTION,SAVE,ADD,DELETE,EDIT

INTEGER CONVER,REPLAC,LIST,DONE

COMMON /MISC/ERFLAG,FERR,KARD(14),ACTION(14)

INTEGER ERFLAG,FERR,ACTION,PRTFIL

EQUIVALENCE (PRTFIL,FERR)

COMMON /FILES/ BASIC,MTAPE,FINAL,S1,S2

INTEGER BASIC,MTAPE,FINAL,S1,S2

COMMON /REST/TABLE,USES,FILE,END,DECK,DDECK,BLANK ,BATCH

INTEGER TABLE,USES,FILE,END,DECK,DDECK,BLANK,BATCH

DATA INPUT/5/,CREATE/6HCREATE/,OPTION/6HOPTION/,SAVE/4HSAVE/

DATA ADD/3HADD/,DELETE/6HDELETE/,EDIT/4HEDIT/

DATA CONVER/6HCONVER/,REPLAC/6HREPLAC/,LIST/4HLIST/,DONE/4HDONE/

DATA CONTEN/6HCONTEN/,GENEOL/6HGENEOL/

DATA USE/3HUSE/

DATA IU/5,0,0/

DATA IR/2,0,0/

DATA IFLAG/0/

MAIN PROGRAM

PROGRAMMER - S. WRAY

RESERVED FILES

TAPE 1 - INPUT DATA BANK

TAPE 4 - OUTPUT DATA BANK

TAPE 12 - DORCA DATA DECK

TAPE 13 - MOD DECKS -INPUT

TAPE 14 - BASIC DECK

TAPE 21 - INPUT BASIC DECKS

TAPE 24 - MOD DECKS - OUTPUT

DORMAN 3

DORMAN13

DORMAN14

DORMAN15

DORMAN16

MISC 2

MISC 3

MISC 4

FILES 2

FILES 3

REST 2

REST 3

DORMAN20

DORMAN21

DORMAN22

DORMAN23

DORMAN24

DORMAN25

DORMAN26

DORMAN27

DORMAN28

DORMAN29

DORMAN30

DORMAN31

DORMAN32

DORMAN33

DORMAN34

DORMAN35

DORMAN36

DORMAN37

DORMAN38

DORMAN39

DORMAN40

DORMAN41

DORMAN42

DORMAN43

WRITE(PRTFIL,5)	DORMAN48
5 FORMAT(21H1 -- BEGIN DORMAN --//34H DO YOU WISH TO CREATE A DATA	DORMAN49
1 FILE/22H OR TO USE SUCH A FILE/20H ENTER CREATE OR USE)	DORMAN50
10 WRITE(PRTFIL,15)	DORMAN51
15 FORMAT(24H)ENTER OPTION REQUEST -)	DORMAN52
CALL INTBUF(IU,IR)	DORMAN53
REWIND BASIC	DORMAN54
CALL INCD (KARD,INPUT)	DORMAN55
IF(IFLAG.NE.0) GO TO 17	DORMAN56
IF(KARD(1).EQ.CREATE) GO TO 20	DORMAN57
17 IF(KARD(1).EQ.1HU) GO TO 30	DORMAN58
IF(KARD(1).EQ.USE) GO TO 30	DORMAN59
IF(IFLAG.EQ.0) GO TO 18	DORMAN60
IF(KARD(1).EQ.1HO) GO TO 25	DORMAN61
IF(KARD(1).EQ.1HS) GO TO 40	DORMAN62
IF(KARD(1).EQ.1HA) GO TO 35	DORMAN63
IF(KARD(1).EQ.1HD) GO TO 45	DORMAN64
IF(KARD(1).EQ.1HE) GO TO 50	DORMAN65
IF(KARD(1).EQ.1HC) GO TO 55	DORMAN66
IF(KARD(1).EQ.1HR) GO TO 60	DORMAN67
IF(KARD(1).EQ.1HL) GO TO 65	DORMAN68
IF(KARD(1).EQ.OPTION) GO TO 25	DORMAN69
IF(KARD(1).EQ.SAVE) GO TO 40	DORMAN70
IF(KARD(1).EQ.ADD) GO TO 35	DORMAN71
IF(KARD(1).EQ.DELETE) GO TO 45	DORMAN72
IF(KARD(1).EQ.EDIT) GO TO 50	DORMAN73
IF(KARD(1).EQ.CONVER) GO TO 55	DORMAN74
IF(KARD(1).EQ.REPLAC) GO TO 60	DORMAN75
IF(KARD(1).EQ.LIST) GO TO 65	DORMAN76
IF(KARD(1).EQ.DONE) GO TO 70	DORMAN77
IF(KARD(1).EQ.SHATCH) GO TO 80	DORMAN78
18 CONTINUE	DORMAN79
WRITE (PRTFIL,16)	DORMAN80
16 FORMAT(40H COMMAND NOT UNDERSTOOD - PLEASE RETRY)	DORMAN81
IF(ERFLAG.NE.0) CALL TERM	DORMAN82
ERFLAG=1	DORMAN83

```

        GO TO 100
65  CONTINUE
    CALL LISTER
    GO TO 100
70  CONTINUE
    CALL TERM
    GO TO 100
20  CONTINUE
    BASIC = 2
    CALL INCR
    IFLAG = 1
    GO TO 100
25  CONTINUE
    CALL OPT
    GO TO 100
30  CONTINUE
    CALL USER (IFLAG)
    IFLAG = 1
    GO TO 100
40  CONTINUE
    CALL SAVER
    GO TO 100
35  CONTINUE
    CALL ADDER
    GO TO 100
45  CONTINUE
    CALL DELET
    GO TO 100
50  CONTINUE
    CALL EDITER
    GO TO 100
55  CONTINUE
    CALL CONV
    GO TO 100
60  CONTINUE
    CALL REPL

```

```

DORMAN84
UORMAN85
DORMAN86
DORMAN87
DORMAN88
DORMAN89
DORMAN90
UORMAN91
DORMAN92
DORMAN93
UORMAN94
DORMAN95
DORMAN96
DORMAN97
DORMAN98
UORMAN99
DORMA100
DORMA101
DORMA102
DORMA103
DORMA104
DORMA105
DORMA106
DORMA107
DORMA108
DORMA109
DORMA110
DORMA111
DORMA112
DORMA113
DORMA114
DORMA115
DORMA116
DORMA117
DORMA118
DORMA119

```


GO TO 100	DORMA120
80 CONTINUE	DORMA121
BATCH = 5	DORMA122
GO TO 100	DORMA123
100 ERFLAG=0	DORMA124
GO TO 10	DORMA127
END	DORMA128
*ELT,I DORMAN,ADD	ADD 3
SUBROUTINE ADD(IN1,IN2,OUT)	ADD 5
C ADD A DECK (WHICH RESIDES ON FILE IN2) TO THE DATA FILE (IN1) AND	ADD 6
C PUT EXPANDED DATA ON FILE OUT.	ADD 7
C BY B.J. GOLD	ADD 8
C	ADD 9
COMMON /MISC/ERFLAG,FERR,KARD(14),ACTION(14)	MISC 2
INTEGER ERFLAG,FERR,ACTION,PRTFIL	MISC 3
EQUIVALENCE (PRTFIL,FERR)	MISC 4
INTEGER OUT	ADD 11
INTEGER TYPE, FLAG, TEMP(14)	ADD 12
COMMON /REST/TABLE,USES,FILE,END,DECK,DDECK,BLANK ,BATCH	REST 2
INTEGER TABLE,USES,FILE,END,DECK,DDECK,BLANK,BATCH	REST 3
C	ADD 14
C INITIALIZE FILES. READ FIRST CARD OF NEW DECK AND VERSION CARD FROM	ADD 15
C DATA FILE.	ADD 16
C	ADD 17
FLAG = 1	ADD 18
REWIND IN1	ADD 19
REWIND OUT	ADD 20
ERFLAG = 0	ADD 21
CALL INGD(KARD,IN2)	ADD 22
KOUNT2 = 1	ADD 23
IF (ERFLAG.NE.0) GO TO 100	ADD 24
TYPE = 5HBASIC	ADD 25
IF (KARD(4).EQ.USES) TYPE = 3HMOD	ADD 26
IF (KARD(1).NE.DDECK) GO TO 110	ADD 27
CALL EXTRAN (KARD(7),KARD(9))	ADD 28
WRITE (PRTFIL,10) KARD(2), KARD(3)	ADD 29

10	FORMAT(16H0ADD DECK NAMED ,2A6,13H TO DATA FILE)	ADD	30
	CALL OUTCD (KARD,PRTFIL)	ADD	31
	IF ((KARD(2).EQ.KARD(5)).AND.(KARD(3).EQ.KARD(6))) GO TO 150	ADD	32
	CALL INCD(TEMP,IN1)	ADD	33
	IF (ERFLAG.NE.0) GO TO 120	ADD	34
	KOUNT1 = 1	ADD	35
	CALL OUTCD(TEMP,OUT)	ADD	36
C		ADD	37
C	INSERT NEW ENTRY INTO TABLE OF CONTENTS AT BEGINNING IF IT IS A MOD	ADD	38
C	DECK, AT END IF A BASIC DECK.	ADD	39
C	VERIFY THAT NEW DECK NAME IS NOT ALREADY IN TABLE OF CONTENTS.	ADD	40
C	IF NEW DECK IS A MOD DECK, VERIFY THAT THE DECK IT USES IS ALREADY	ADD	41
C	ON THE DATA FILE.	ADD	42
C		ADD	43
	FLAG = 2	ADD	44
	IF (TYPE.EQ.3HMOD) CALL OUTCD(KARD,OUT)	ADD	45
	NEED = 0	ADD	46
	LASTM = 0	ADD	47
	NTC = 0	ADD	48
30	CALL INCD(TEMP,IN1)	ADD	49
	IF (ERFLAG.NE.0) GO TO 120	ADD	50
	KOUNT1 = KOUNT1+1	ADD	51
	IF (TEMP(1).EQ.END) GO TO 40	ADD	52
	CALL OUTCD(TEMP,OUT)	ADD	53
	NTC = NTC+1	ADD	54
	IF (TEMP(4).EQ.USES) LASTM = NTC	ADD	55
	IF ((KARD(2).EQ.TEMP(2)).AND.(KARD(3).EQ.TEMP(3))) GO TO 130	ADD	56
	IF ((KARD(5).EQ.TEMP(2)).AND.(KARD(6).EQ.TEMP(3))) NEED = NTC	ADD	57
	GO TO 30	ADD	58
40	IF (TYPE.EQ.5HBASIC) CALL OUTCD(KARD,OUT)	ADD	59
	CALL OUTCD(TEMP,OUT)	ADD	60
	IF ((NEED.EQ.0).AND.(TYPE.EQ.3HMOD)) GO TO 140	ADD	61
C		ADD	62
C	COPY ALL MOD DECKS FROM IN1 TO OUT	ADD	63
C		ADD	64
	FLAG = 3	ADD	65

NDECK = 0	ADD	66
50 CALL INCD(TEMP,IN1)	ADD	67
IF (ERFLAG.NE.0) GO TO 120	ADD	68
KOUNT1 = KOUNT1 + 1	ADD	69
CALL OUTCD(TEMP,OUT)	ADD	70
IF (TEMP(1).NE.END) GO TO 50	ADD	71
NDECK = NDECK+1	ADD	72
IF (NDECK.LT.LASTM) GO TO 50	ADD	73
C	ADD	74
C TRANSFER NEW DECK ONTO OUT FILE	ADD	75
C	ADD	76
FLAG = 4	ADD	77
CALL OUTCD(KARD,OUT)	ADD	78
60 CALL INCD(KARD,IN2)	ADD	79
IF (ERFLAG.NE.0) GO TO 100	ADD	80
KOUNT2 = KOUNT2+1	ADD	81
CALL OUTCD(KARD,OUT)	ADD	82
IF (KARD(1).NE.END) GO TO 60	ADD	83
C	ADD	84
C COPY REST OF DATA FILE, INCLUDING \$END OF FILE CARD, ONTO OUT FILE	ADD	85
C	ADD	86
FLAG = 5	ADD	87
80 CALL INCD(TEMP,IN1)	ADD	88
IF (ERFLAG.NE.0) GO TO 120	ADD	89
KOUNT1 = KOUNT1+1	ADD	90
CALL OUTCD(TEMP,OUT)	ADD	91
IF (TEMP(1).NE.END) GO TO 80	ADD	92
IF (TEMP(2).NE.6HF FILE) GO TO 80	ADD	93
RETURN	ADD	94
C	ADD	95
C ERROR STOPS	ADD	96
C	ADD	97
100 WRITE (FERR,105)	ADD	98
105 FORMAT(23H0ERROR READING FILE IN2)	ADD	99
GO TO 200	ADD	100
110 WRITE (FERR,115)	ADD	101

115	FORMAT(37H0FIRST CARD ON FILE IN2 IS NOT SDECK)	ADD	102
	GO TO 200	ADD	103
120	WRITE (FERR,125)	ADD	104
125	FORMAT(23H0ERROR READING FILE IN1)	ADD	105
	GO TO 200	ADD	106
130	WRITE (FERR,135) KARD(2), KARD(3)	ADD	107
135	FORMAT(45H0NAME CONFLICT IN ADDING NEW DECK TO FILE IN1/ * 6H NAME ,2A6,36H ALREADY EXISTS IN TABLE OF CONTENTS)	ADD	108
	GO TO 200	ADD	109
140	WRITE (FERR,145) (KARD(I), I=1,6)	ADD	110
145	FORMAT(52H0DECK TO BE ADDED USES DECK NOT IN TABLE OF CONTENTS/ * 5X,6A6)	ADD	111
	GO TO 200	ADD	112
150	WRITE (FERR,155)	ADD	113
155	FORMAT(17H0DECK USES ITSELF)	ADD	114
C		ADD	115
200	ERFLAG = 1	ADD	116
C		ADD	117
C	ADVANCE FILE IN2 TO END OF CURRENT DECK	ADD	118
C		ADD	119
	ERFLAG = 0	ADD	120
210	IF (KARD(1).EQ.END) GO TO 220	ADD	121
	CALL INCD(KARD,IN2)	ADD	122
	KOUNT2 = KOUNT2 + 1	ADD	123
	IF (ERFLAG.EQ.0) GO TO 210	ADD	124
	WRITE (FERR,165)	ADD	125
220	ERFLAG = 1	ADD	126
	RETURN	ADD	127
	END	ADD	128
#ELT,I	DORMAN.ADDER	ADD	129
	SUBROUTINE ADDER	ADD	130
C		ADDER	3
C	ADD CONTRULER	ADDER	5
C		ADDER	6
C		ADDER	7
C	PROGRAMMER - S. WRAY	ADDER	8
C		ADDER	9
		ADDER	10

COMMON /MISC/ERFLAG,FERR,KARD(14),ACTION(14)	MISC	2
INTEGER ERFLAG,FERR,ACTION,PRTFIL	MISC	3
EQUIVALENCE (PRTFIL,FERR)	MISC	4
DATA IX/0/	ADDER	12
IF(IX.EQ.0) REWIND 21	ADDER	13
IX = 1	ADDER	14
1 WRITE (PRTFIL,5)	ADDER	15
5 FORMAT(12H ENTER INPUT/12H OR BASIC/15H OR MOD DECK/9H READY	ADDER	16
* -)	ADDER	17
CALL INCO(KARD,5)	ADDER	18
IF(KARD(1).EQ.5HBASIC) GO TO 10	ADDER	19
IF(KARD(1).EQ.6HMOD DE) GO TO 15	ADDER	20
IF(KARD(1).NE.5HINPUT) GO TO 1	ADDER	21
CALL ADDX(21)	ADDER	22
RETURN	ADDER	23
10 CONTINUE	ADDER	24
REWIND 14	ADDER	25
CALL ADDX (14)	ADDER	26
RETURN	ADDER	27
15 CONTINUE	ADDER	28
WRITE (PRTFIL,20)	ADDER	29
20 FORMAT(*2H IS MOD DECK GENERATED OR INPUT ON TAPE 11/9H READY -)	ADDER	30
CALL INCO (KARD,5)	ADDER	31
M1 = 13	ADDER	32
IF(KARD(1).EQ.5HINPUT) M1 = 11	ADDER	33
REWIND M1	ADDER	34
CALL ADDX(M1)	ADDER	35
RETURN	ADDER	36
END	ADDER	37
*ELT,I DORMAN.ADDX	ADUX	3
SUBROUTINE ADDX (INDKF)	ADUX	5
C	ADUX	6
C CONTROLS ADD OPERATION	ADUX	7
C	ADUX	8
C PROGRAMMER - S. WRAY	ADUX	9
C	ADUX	10

COMMON /REST/TABLE,USES,FILE,END,DECK,DDECK,BLANK ,BATCH	REST	2
INTEGER TABLE,USES,FILE,END,DECK,DDECK,BLANK,BATCH	REST	3
COMMON /FILES/ BASIC,MTAPE,FINAL,S1,S2	FILES	2
INTEGER BASIC,MTAPE,FINAL,S1,S2	FILES	3
COMMON /MISC/ERFLAG,FERR,KARD(14),ACTION(14)	MISC	2
INTEGER ERFLAG,FERR,ACTION,PRTFIL	MISC	3
EQUIVALENCE (PRTFIL,FERR)	MISC	4
INTEGER OUTOK,GU,DONE,IU(3),IR(3)	ADDX	14
DATA IR/2,2,2/	ADDX	15
DATA GU/2HGO/,DONE/4HDONE/	ADDX	16
C	ADDX	17
C INPUT DECKS ARE ON TAPE21, TEMP OUT FILE IS TAPE22	ADDX	18
C	ADDX	19
IC = 0	ADDX	20
IX = 1	ADDX	21
OUTOK = 22	ADDX	22
10 CONTINUE	ADDX	23
NO = 0	ADDX	24
IU(1) = 5	ADDX	25
IU(2) = INDKF	ADDX	26
IU(3) = OUTOK	ADDX	27
CALL INTBUF(IU,IR)	ADDX	28
CALL INCD (KARD,INDKF)	ADDX	29
IF (ERFLAG.NE.0) GO TO 1000	ADDX	30
12 REWIND OUTOK	ADDX	31
IF (KARD(1).EQ.DDECK) GO TO 20	ADDX	32
WRITE (PRTFIL,11)	ADDX	33
11 FORMAT(27H PLEASE ENTER NAME FOR DECK / 9H READY -)	ADDX	34
CALL INCD(ACTION,5)	ADDX	35
ACTION (3) = ACTION(2)	ADDX	36
ACTION (2) = ACTION(1)	ADDX	37
ACTION (1) = DDECK	ADDX	38
CALL OUTCD (ACTION,OUTOK)	ADDX	39
20 CONTINUE	ADDX	40
IC = IC+1	ADDX	41
NO = 1	ADDX	42

39	WRITE (PRTFIL,40) KARD(2),KARD(3)	ADUX	43
40	FORMAT(14H DECK FOUND - ,2A6/17H ENTER OK OR SKIP/9H READY -)	ADUX	44
	CALL INCD (ACTION,5)	ADUX	45
	IF(ACTION(1).EQ.2HOK) GO TO 45	ADUX	46
	IF(ACTION(1).NE.4HSKIP) GO TO 39	ADUX	47
41	CALL INCD (KARD,INDKF)	ADUX	48
	IF(ERFLAG.NE.0) RETURN	ADUX	49
	IF(KARD(1).NE.DLECK) GO TO 41	ADUX	50
	GO TO 20	ADUX	51
45	CONTINUE	ADUX	52
	CALL OUTCD(KARD,OUTOK)	ADUX	53
	IC = IC + 1	ADUX	54
21	CALL INCD (KARD,INDKF)	ADUX	55
	ACTION(2)=KARD(2)	ADUX	56
	IF(ERFLAG.NE.0) RETURN	ADUX	57
	IF(KARD(1).NE.END) GO TO 45	ADUX	58
23	CONTINUE	ADUX	59
	DO 22 I = 1,14	ADUX	60
22	KARD(I) = BLANK	ADUX	61
	KARD(1) = END	ADUX	62
	KARD(2) = DECK	ADUX	63
	CALL OUTCD(KARD,OUTOK)	ADUX	64
	KARD(2) = FILE	ADUX	65
	IX = 2	ADUX	66
	IF(ACTION(2).EQ.DECK) GO TO 30	ADUX	67
1000	WRITE (PRTFIL,1001)	ADUX	68
1001	FORMAT(51H EOF FOUND - IF ACCEPTED, ENTER GO, OTHERWISE DONE /9H RA	ADUX	69
	*EADY -)	ADUX	70
	CALL INCD (ACTION,5)	ADUX	71
	IF(ACTION(1).EQ.DONE) RETURN	ADUX	72
	IF(ACTION(1).NE.GO) GO TO 1000	ADUX	73
	ERFLAG = 0	ADUX	74
30	CONTINUE	ADUX	75
	WRITE (PRTFIL,31) IC	ADUX	76
31	FORMAT(16,20H CARUS FOUND IN DECK)	ADUX	77
	IC = 0	ADUX	78

```

      S1 = BASIC + 1
      IF(S1.GT.3) S1 = 2
      REWIND BASIC
      REWIND OUTOK
      REWIND S1
      IU(1) = BASIC
      IU(2) = OUTOK
      IU(3) = S1
      CALL INTBUF(IU,IR)
      CALL ADD(BASIC,OUTOK,S1)
      IF(ERFLAG.NE.0) RETURN
      BASIC = S1
      RETURN
      END
*ELT,I DORMAN,ASSGN
      SUBROUTINE ASSGN (I)
C
C   ISSUE ASG CARD FOR FILE I
C
C   PROGRAMMER - S. WRAY
C
      COMMON /MISC/ERFLAG,FERR,KARD(14),ACTION(14)
      INTEGER ERFLAG,FERR,ACTION,PRTFIL
      EQUIVALENCE (PRTFIL,FERR)
      WRITE (PRTFIL,5) I
      5 FORMAT(28H WHAT IS YOUR NAME FOR TAPE ,13/9H READY - )
      CALL INCD (ACTION,5)
      IF(ERFLAG.NE.0) RETURN
C
C   ISSUE ASG AND USE CARDS
C
      RETURN
      END
*ELT,I DORMAN,CKSYN
      SUBROUTINE CKSYN (ICD1,ICD2,N1,N2)
C

```

```

      ADDX 79
      ADDX 80
      ADDX 81
      ADDX 82
      ADDX 83
      ADDX 84
      ADDX 85
      ADDX 86
      ADDX 87
      ADDX 88
      ADDX 89
      ADDX 90
      ADDX 91
      ADDX 92
      ASSGN 4
      ASSGN 6
      ASSGN 7
      ASSGN 8
      ASSGN 9
      ASSGN 10
      ASSGN 11
      MISC 2
      MISC 3
      MISC 4
      ASSGN 13
      ASSGN 14
      ASSGN 15
      ASSGN 16
      ASSGN 17
      ASSGN 18
      ASSGN 19
      ASSGN 20
      ASSGN 21
      CKSYN 3
      CKSYN 5
      CKSYN 6

```


C	ICD1 IS CARD IN BUFFER1	CKSYN 7
C	ICD2 IS CARD IN BUFFER2	CKSYN 8
C	N1 IS INDEX TO PROPER SYNC CARD	CKSYN 9
C	N2 = 0, DO NOT INCREMENT N1	CKSYN 10
C	N2 = 1, OK TO INCREMENT N1	CKSYN 11
	COMMON /BFRS/	CKSYN 12
	* ISYN(2,20)	CKSYN 13
C		CKSYN 14
C	IF ICD1 OR ICD2 IS GE TO SYNC CARD THEN	CKSYN 15
C	RESET TO SYNC CARD VALUES	CKSYN 16
C		CKSYN 17
C	PROGRAMMER: VOIT	CKSYN 18
C		CKSYN 19
	ISYN1 = N1	CKSYN 20
	ICD1 = ISYN(1,ISYN1)	CKSYN 21
	ICD2 = ISYN(2,ISYN1)	CKSYN 22
	IF (N2.EQ.0) RETURN	CKSYN 23
	N1 = N1+1	CKSYN 24
	RETURN	CKSYN 25
	END	CKSYN 26
*ELT,I	DORMAN.CLOSE	CLOSE 4
	SUBROUTINE CLOSE (I)	CLOSE 5
	ENDFILE I	CLOSE 7
	REWIND I	CLOSE 8
	RETURN	CLOSE 9
	END	CLOSE 10
*ELT,I	DORMAN.COUNT	COUNT 3
	SUBROUTINE COUNT(NFILE)	COUNT 5
	COMMON /REST/TABLE,USES,FILE,END,DECK,DDECK,BLANK ,BATCH	REST 2
	INTEGER TABLE,USES,FILE,END,DECK,DDECK,BLANK,BATCH	REST 3
	COMMON /MISC/ERFLAG,FERR,KARD(14),ACTION(14)	MISC 2
	INTEGER ERFLAG,FERR,ACTION,PRTFIL	MISC 3
	EQUIVALENCE (PRTFIL,FERR)	MISC 4
C		COUNT 8
C	PROGRAMMER - S. WRAY	COUNT 9
C		COUNT 10

REWIND NFILE	COUNT 11
ACTION (1) = 6HNO NAM	COUNT 12
ACTION (2) = 6HE	COUNT 13
DO 1 I=3,6	COUNT 14
1 ACTION(I) = BLANK	COUNT 15
I=1	COUNT 16
CALL INCD(KARD,NFILE)	COUNT 17
IF(ERFLAG.NE.0) RETURN	COUNT 18
IF(KARD(1).NE.DDECK) GO TO 5	COUNT 19
ACTION (1) = KARD (1)	COUNT 20
ACTION (2) = KARD (2)	COUNT 21
ACTION (3) = KARD (3)	COUNT 22
ACTION (4) = KARD (4)	COUNT 23
ACTION (5) = KARD (5)	COUNT 24
ACTION (6) = KARD (6)	COUNT 25
5 CALL INCD (KARD,NFILE)	COUNT 26
IF(ERFLAG.NE.0) GO TO 10	COUNT 27
I=I+1	COUNT 28
IF(KARD(1).NE.END) GO TO 5	COUNT 29
IF(KARD(2).NE.FILE) GO TO 5	COUNT 30
10 WRITE (PRTFIL,15) NFILE,(ACTION(LL),LL=1,6),I	COUNT 31
15 FORMAT(5H0TAPE,I3,6H WITH ,6A6,9H CONTAINS,I7,6H CARDS/)	COUNT 32
RETURN	COUNT 33
END	COUNT 34
*ELT,I JORMAN.DELCD	DELCD 3
SUBROUTINE DELCD (ISP)	DELCD 5
C	DELCD 6
C ISP IS THE SYNC CARD	DELCD 7
C	DELCD 8
C WRITE DELETE CARD	DELCD 9
C AND POSITION BUFFER IF REQUIRED	DELCD 10
C	DELCD 11
COMMON /BFRS/ ISYN(2,20),ISYN1,IT1,IT2,IT3,FULL,CN1,CN2,LIN1,LIN2	BFRS1 2
* ,NWBUF,NB1,NB2,ICN1,ICN2	BFRS1 3
COMMON /BFRS/ BUF1(14,50),BUF2(14,50),AAA(700)	BFRS1 4
INTEGER CN1,CN2,BUF1,BUF2	BFRS1 5

LOGICAL FULL	5FRS1	0
INTEGER FIN1,FIN2	DELCD	13
EQUIVALENCE (FOUT,IT3)	DELCD	14
EQUIVALENCE (FIN1,IT1)	DELCD	15
EQUIVALENCE (FIN2,IT2)	DELCD	16
INTEGER FOUT	DELCD	17
KK = ISP-2	DELCD	18
KKK = LIN1 - 1	DELCD	19
WRITE(FOUT,81) KKK,KK,	DELCD	20
C	DELCD	21
C	DELCD	22
C	DELCD	23
C	DELCD	24
NCARDS = KK - KKK + 1	DELCD	25
CN1 = CN1 + NCARDS	DELCD	26
LIN1 = LIN1 + NCARDS	DELCD	27
IF(CN1.LT.NB1) RETURN	DELCD	28
ISP = LIN1	DELCD	29
LIN1 = LIN1 - NCARDS	DELCD	30
CALL RESTOR(BUF1,ISP,NB1,FIN1,LIN1)	GWT13	1
CN1 = 1	DELCD	32
LIN1 = ISP	DELCD	33
RETURN	DELCD	34
81 FORMAT (7H\$DELETE,I13,I10)	DELCD	35
END	DELCD	36
*ELT,I DORMAN.DELET	DELET	3
SUBROUTINE DELET	DELET	5
C	DELET	6
C	DELET	7
C	DELET	8
C	DELET	9
C	DELET	10
DIMENSION IU(3),IR(3)	DELET	11
COMMON /MISC/ERFLAG,FERR,KARD(14),ACTION(14)	MISC	2
INTEGER ERFLAG,FERR,ACTION,PRTFIL	MISC	3
EQUIVALENCE (PRTFIL,FERR)	MISC	4

```

COMMON /FILES/ BASIC,MTAPE,FINAL,S1,S2
INTEGER BASIC,MTAPE,FINAL,S1,S2
DATA IR/2,2,2/
WRITE (PRTFIL,5)
5 FORMAT(27H WHAT DECK IS TO BE DELETED/9H READY - )
S1 = BASIC +1
IF(S1.GT.3) S1=2
IU(1) = 5
IU(2) = BASIC
IU(3) = S1
CALL INTBUF(IU,IR)
CALL INCU (ACTION,5)
CALL DELETE (ACTION,BASIC,S1)
IF(ERFLAG.NE.0) RETURN
BASIC=S1
RETURN
END
*ELT,I DORMAN.DELETE
SUBROUTINE DELETE(NAME,IN1,OUT)
C
C COPY DATA FROM FILE IN1 TO FILE OUT WITH NAMED DECK DELETED.
C PROGRAMMER - B. GOLD
C
COMMON /MISC/ERFLAG,FERR,KARD(14),ACTION(14)
INTEGER ERFLAG,FERR,ACTION,PRTFIL
EQUIVALENCE (PRTFIL,FERR)
INTEGER OUT
INTEGER FLAG, NAME(2), TEMP(14)
EQUIVALENCE (KARD(1),TEMP(1))
COMMON /REST/TABLE,USES,FILE,END,DECK,DDECK,BLANK ,BATCH
INTEGER TABLE,USES,FILE,END,DECK,DDECK,BLANK,BATCH
C
C INITIALIZE FILES AND COPY VERSION CARD.
C
FLAG = 1
REWIND IN1

```

```

FILES 2
FILES 3
DELET 14
DELET 15
DELET 16
DELET 17
DELET 18
DELET 19
DELET 20
DELET 21
DELET 22
DELET 23
DELET 24
DELET 25
DELET 26
DELET 27
DELET 28
DELETE 3
DELETE 5
DELETE 6
DELETE 7
DELETE 8
DELETE 9
MISC 2
MISC 3
MISC 4
DELETE11
DELETE12
DELETE13
REST 2
REST 3
DELETE15
DELETE16
DELETE17
DELETE18
DELETE19

```

```

REWIND OUT
ERFLAG = 0
WRITE (PRTFIL,10) NAME
10 FORMAT(19HDELETE DECK NAMED ,2A6,15H FROM DATA FILE)
CALL INCD(TEMP,IN1)
IF (ERFLAG.NE.0) GO TO 120
KOUNT1 = 1
CALL OUTCD(TEMP,OUT)
C
C COPY TABLE OF CONTENTS. VERIFY THAT DECK TO BE DELETED IS IN TABLE
C OF CONTENTS AND NOT REQUIRED BY ANY OTHER DECK.
C
FLAG = 2
INDEX = 0
NTC = 0
30 CALL INCD(TEMP,IN1)
IF (ERFLAG.NE.0) GO TO 120
KOUNT1 = KOUNT1+1
IF (TEMP(1).EQ.END) GO TO 40
NTC = NTC+1
IF ((TEMP(2).EQ.NAME(1)).AND.(TEMP(3).EQ.NAME(2))) GO TO 35
IF ((TEMP(5).EQ.NAME(1)).AND.(TEMP(6).EQ.NAME(2))) GO TO 140
CALL OUTCD(TEMP,OUT)
GO TO 30
35 INDEX = NTC
GO TO 30
40 CALL OUTCD(TEMP,OUT)
IF (INDEX.EQ.0) GO TO 130
C
C COPY ALL DECKS FROM IN1 TO OUT UNTIL REQUIRED DECK IS FOUND.
C
FLAG = 3
50 CALL INCD(TEMP,IN1)
IF (ERFLAG.NE.0) GO TO 120
KOUNT1 = KOUNT1 + 1
IF ((TEMP(1).EQ.DDECK).AND.(TEMP(2).EQ.NAME(1)))

```

```

DELETE20
DELETE21
DELETE22
DELETE23
DELETE24
DELETE25
DELETE26
DELETE27
DELETE28
DELETE29
DELETE30
DELETE31
DELETE32
DELETE33
DELETE34
DELETE35
DELETE36
DELETE37
DELETE38
DELETE39
DELETE40
DELETE41
DELETE42
DELETE43
DELETE44
DELETE45
DELETE46
DELETE47
DELETE48
DELETE49
DELETE50
DELETE51
DELETE52
DELETE53
DELETE54
DELETE55

```

	*	.AND.(TEMP(3).EQ.NAME(2)))	GO TO 55	DELETE56
		CALL OUTCD(TEMP,OUT)		DELETE57
		GO TO 50		DELETE58
C				DELETE59
C		SPACE PAST OLD DECK		DELETE60
C				DELETE61
	55	FLAG = +		DELETE62
		KOUNT2 = KOUNT1		DELETE63
	60	CALL INCD(TEMP,IN1)		DELETE64
		IF (ERFLAG.NE.0) GO TO 120		DELETE65
		KOUNT1 = KOUNT1 + 1		DELETE66
		IF (TEMP(1).NE.END) GO TO 60		DELETE67
		KOUNT2 = KOUNT1-KOUNT2+1		DELETE68
		WRITE (PRIFIL,70) KOUNT2		DELETE69
	70	FORMAT(/I10,14H CARDS DELETED)		DELETE70
C				DELETE71
C		COPY REST OF DATA FILE, INCLUDING SEND OF FILE CARD, ONTO OUT FILE		DELETE72
C				DELETE73
		FLAG = 5		DELETE74
	80	CALL INCD(TEMP,IN1)		DELETE75
		IF (ERFLAG.NE.0) GO TO 120		DELETE76
		KOUNT1 = KOUNT1+1		DELETE77
		CALL OUTCD(TEMP,OUT)		DELETE78
		IF (TEMP(1).NE.END) GO TO 80		DELETE79
		IF (TEMP(2).NE.6HF FILE) GO TO 80		DELETE80
		RETURN		DELETE81
C				DELETE82
C		ERROR STOPS		DELETE83
C				DELETE84
	120	WRITE (FERR,125)		DELETE85
	125	FORMAT(23H0ERROR READING FILE IN1)		DELETE86
		GO TO 200		DELETE87
	130	WRITE (FERR,135) NAME		DELETE88
	135	FORMAT(22H0DECK TO BE DELETED (,2A6,36H) IS NOT LISTED IN TABLE		DELETE89
		*F CONTENTS)		DELETE90
		GO TO 200		DELETE91

140 WRITE (FERR,145) NAME, TEMP(2), TEMP(3)	DELETE92
145 FORMAT(21HODECK TO BE DELETED (2A6,22H) IS REQUIRED BY DECK 2A6)	DELETE93
C	DELETE94
200 ERFLAG = 1	DELETE95
RETURN	DELETE96
END	DELETE97
#ELT,I DORMAN.DIFDEC	DIFDEC03
SUBROUTINE DIFDEC (FIN1,FIN2,FOUT,FSYN)	DIFDEC05
C	DIFDEC06
FSYN WILL CONTAIN THE SYNC CARDS	DIFDEC07
C THE SYNC SEARCH WILL BE ABANDONED IF THE CARD	DIFDEC08
C NUMBER IS GREATER THAN THE SYNC CARD	DIFDEC09
C THEN THE SYNC CARD NUMBERS WILL BE USED	DIFDEC10
C	DIFDEC11
COMMON /BUFFER/MAX	DIFDEC12
COMMON /BFRS/ ISYN(2,20),ISYN1,IT1,IT2,IT3,FULL,CN1,CN2,LIN1,LIN2	BFRS1 2
* ,NWBUF,NB1,NB2,ICN1,ICN2	BFRS1 3
COMMON /BFRS/ BUF1(14,50),BUF2(14,50),AAA(700)	BFRS1 4
INTEGER CN1,CN2,BUF1,BUF2	BFRS1 5
LOGICAL FULL	BFRS1 6
INTEGER FIN1,FIN2,FOUT	DIFDEC14
INTEGER FSYN	DIFDEC15
C	DIFDEC16
C THIS ROUTINE DETERMINES THE DIFFERENCE BETWEEN DECKS ON TAPES FIN1	DIFDEC17
C	DIFDEC18
C AND FIN2. THE DIFFERENCE IS EXPRESSED AS A MOD DECK TO CONVERT	DIFDEC19
C	DIFDEC20
C THE DECK ON FIN1 TO THE DECK ON FIN2. THE MOD DECK IS WRITTEN	DIFDEC21
C	DIFDEC22
C ON FOUT.	DIFDEC23
C	DIFDEC24
DIMENSION IUNT(3),IRW(3)	DIFDEC25
INTEGER COUT(14)	DIFDEC26
NWBUF = 50	DIFDEC27
MAX= NWBUF	DIFDEC28
IT1 = FIN1	DIFDEC29

```

      IT2 = FIN2
      IT3 = FOUT
C
C      READ SYNC CARDS
C
      IT4 = FSYN
      CALL SYNCDS(IT4)
C
C      INITIALIZE UNITS
C
      ISYN1 = 1
      IUNT(1)=FIN1
      IUNT(2)=FIN2
      IUNT(3)=FOUT
      IRW(1) = 0
      IRW(2) = 0
      IRW(3) = 1
      CALL INTBUF (IUNT,IRW)
      CALL FILBUF(1,NB1,NWBUF,BUF1,FIN1)
      CALL FILBUF(1,NB2,NWBUF,BUF2,FIN2)
      FULL =.TRUE.
      CN1 = 1
      CN2 = 1
      LIN1 = 1
      LIN2 = 1
      DO 5 I=1,14
5  COUT(I)=1H
      COUT(1) = BUF1(1,1)
      COUT(2) = BUF2(2,1)
      COUT(3) = BUF2(3,1)
      COUT(4) = 6H USES
      COUT(5) = BUF1(2,1)
      COUT(6) = BUF1(3,1)
      CALL LEXTRAN (COUT(7),COUT(9))
      CALL OUTCD (COUT,6)
      CALL OUTCD (COUT,FOUT)

```

```

DIFDEC30
DIFDEC31
DIFDEC32
DIFDEC33
DIFDEC34
DIFDEC35
DIFDEC36
DIFDEC37
DIFDEC38
DIFDEC39
DIFDEC40
DIFDEC41
DIFDEC42
DIFDEC43
DIFDEC44
DIFDEC45
DIFDEC46
DIFDEC47
DIFDEC48
DIFDEC49
DIFDEC50
DIFDEC51
DIFDEC52
DIFDEC53
DIFDEC54
DIFDEC55
DIFDEC56
DIFDEC57
DIFDEC58
DIFDEC59
DIFDEC60
DIFDEC61
DIFDEC62
DIFDEC63
DIFDEC64
DIFDEC65

```


10	CN1 = CN1+1	DIFDEC66
	CN2 = CN2+1	DIFDEC67
	LIN2=LIN2+1	DIFDEC68
	LIN1=LIN1+1	DIFDEC69
12	CONTINUE	DIFDEC70
	IF(CN1.GT.NB1) GO TO 100	DIFDEC71
	IF(CN2.GT.NB2) GO TO 150	DIFDEC72
	DO 15 I=1,14	DIFDEC73
	IF(BUF1(I,CN1).NE.BUF2(I,CN2)) GO TO 20	DIFDEC74
15	CONTINUE	DIFDEC75
	FULL = .FALSE.	DIFDEC76
	GO TO 10	DIFDEC77
20	CONTINUE	DIFDEC78
C		DIFDEC79
C	CHECK SYNC CARDS	DIFDEC80
C	PRIOR TO SEARCH FOR MATCH	DIFDEC81
C		DIFDEC82
	N2=0	DIFDEC83
	CALL CKSYN(ICN1,ICN2,ISYN1,N2)	DIFDEC84
500	CONTINUE	DIFDEC85
	IF ((LIN1.LT.ICN1).AND.(LIN2.EQ.ICN2)) GO TO 600	DIFDEC86
	IF ((LIN1.EQ.ICN1).AND.(LIN2.LT.ICN2)) GO TO 620	DIFDEC87
	IF((LIN1.EQ.ICN1).AND.(LIN2.EQ.ICN2))GO TO 630	DIFDEC88
	IF (LIN1.GT.ICN1) GO TO 630	DIFDEC89
	IF (LIN2.GT.ICN2) GO TO 630	DIFDEC90
	GO TO 640	DIFDEC91
C		DIFDEC92
C	DELETE	DIFDEC93
C		DIFDEC94
600	CALL DELCD (ICN1-1)	DIFDEC95
	GO TO 10	DIFDEC96
C		DIFDEC97
C	INSERT	DIFDEC98
C		DIFDEC99
620	CALL INSERT (ICN2-1,0)	DIFDEC100
	GO TO 10	DIFDEC101

```

C
C      GET NEXT SYNC CARDS
C
630  N2=1
      CALL CKSYN(ICN1,ICN2,ISYN1,N2)
      GO TO 500
C
C      FIND SYNC
C
640  CONTINUE
      CALL SYNC1
      GO TO 12
100  CONTINUE
      IF(BUF1(1,NB1).EQ.6H$END 0) GO TO 120
      CALL FILBUF(1,NB1,NWBUF,BUF1,FIN1)
      CN1 = 1
      GO TO 12
120  CONTINUE
      IF(CN2.GT.NB2) GO TO 200
      CALL INSERT(NB2-1,0)
      GO TO 200
150  IF(BUF2(1,NB2).EQ.6H$END 0) GO TO 170
      CALL FILBUF(1,NB2,NWBUF,BUF2,FIN2)
      CN2= 1
      GO TO 12
170  CONTINUE
      IF(CN1.GT.NB1) GO TO 200
      CALL DELCD (NB1-1)
200  CONTINUE
      WRITE(FOUT,5000)
5000 FORMAT(12H$END OF DECK/12H$END OF FILE)
      END FILE FOUT
      RETURN
      END
#ELT,I DORMAN,EDITOK
      SUBROUTINE EDITOK(IN,MOD,NOUT)

```

```

DIFDE102
DIFDE103
DIFDE104
DIFDE105
DIFDE106
DIFDE107
DIFDE108
DIFDE109
DIFDE110
DIFDE111
DIFDE112
DIFDE113
DIFDE114
DIFDE115
DIFDE116
DIFDE117
DIFDE118
DIFDE119
DIFDE120
DIFDE121
DIFDE122
DIFDE123
DIFDE124
DIFDE125
DIFDE126
DIFDE127
DIFDE128
DIFDE129
DIFDE130
DIFDE131
DIFDE132
DIFDE133
DIFDE134
DIFDE135
EDITOK 3
EDITOK 5

```

COMMON /MISC/ERFLAG,FERR,KARD(14),ACTION(14)	MISC 2
INTEGER ERFLAG,FERR,ACTION,PRTFIL	MISC 3
EQUIVALENCE (PRTFIL,FERR)	MISC 4
DIMENSION INB(14), MODB(14), NTEM1(14), NTEM2(16)	EDITOK 7
DIMENSION IEM(7,9)	EDITOK 8
DIMENSION NTEM3(16)	EDITOK 9
INTEGER OEND	EDITOK10
DATA IEM /6HEDITOK,6H - ,	EDITOK11
16HDECK N,6HAMES N,6HOT COM,6HPATIBL,6HE ,	EDITOK12
26HEDITOK,6H - ,	EDITOK13
26HNON-NU,6HMERIC ,6HENTRY ,6HIN LIN,6HE NO. ,	EDITOK14
36HEDITOK,6H - ,	EDITOK15
36HLINE N,6HO. IS ,6HOUT OF,6H RANGE,6H ,	EDITOK16
46HEDITOK,6H - ,	EDITOK17
46HLINE N,6HO. IS ,6HOUT OF,6H ORDER,6H ,	EDITOK18
56HEDITOK,6H - ,	EDITOK19
56HMODS R,6HEMAIN ,6H- END ,6HOF IN ,6HLECK ,	EDITOK20
66HEDITOK,6H - ,	EDITOK21
66HCONTIN,6HUATION,6H CARD ,6HOUT OF,6H ORDER,	EDITOK22
76HEDITOK,6H - ,	EDITOK23
76H\$DELET,6HE LINE,6H NOS. ,6HOUT OF,6H ORDER,	EDITOK24
86HEDITOK,6H - ,	EDITOK25
86HNO DAT,6HAMATC,6HH ON 3,6HALTER ,6HCARD ,	EDITOK26
96HEDITOK,6H - ,	EDITOK27
96HCONTIN,6HUATION,6H CARD ,6HNOT EN,6HENTERED /	EDITOK28
	EDITOK29
C PROGRAMMER. - G. W. TIMPSON	EDITOK30
C	EDITOK31
C PRESET VARIABLES	EDITOK32
C	EDITOK33
ERFLAG=0	EDITOK34
I = - 1	EDITOK35
MEND=0	EDITOK36
OEND=0	EDITOK37
IE=0	EDITOK38
NEF=0	EDITOK39

```

C
C   READ H CARD FROM MOD FILE
C
10  NTL=NT
    NT=1
    IF(MEND.EQ.1)GO TO 12
    CALL INCD(MODB,MOD)
    IF(ERFLAG.NE.0)RETURN
    IF(MODB(1).EQ.6H$END 0)GO TO 13
    IF(MODB(1).EQ.6H$INSER)GO TO 14
    IF(MODB(1).EQ.6H$ALTER)GO TO 15
    IF(MODB(1).EQ.6H$DELET) GO TO 16
12  GO TO 20
13  MEND=1
    GO TO 20
14  NT=4
    GO TO 20
15  NT=3
    GO TO 20
16  NT=2
C
C   TEST FOR FIRST TIME
C
20  IF(I.NE.-1) GO TO 22
    NSW=1
    GO TO 600
C
C   TEST FOR PREVIOUS MOD OUT OF RANGE
C
22  IF(IE.EQ.5)GO TO 700
C
C   TEST FOR END OF IN FILE
C
24  IF(CEND.EQ.0)GO TO 30
C
C   TEST FOR CONTINUATION INSERT CARD

```

```

EDITOK40
EDITOK41
EDITOK42
EDITOK43
EDITOK44
EDITOK45
EDITOK46
EDITOK47
EDITOK48
EDITOK49
EDITOK50
EDITOK51
EDITOK52
EDITOK53
EDITOK54
EDITOK55
EDITOK56
EDITOK57
EDITOK58
EDITOK59
EDITOK60
EDITOK61
EDITOK62
EDITOK63
EDITOK64
EDITOK65
EDITOK66
EDITOK67
EDITOK68
EDITOK69
EDITOK70
EDITOK71
EDITOK72
EDITOK73
EDITOK74
EDITOK75

```

```

C      IF(IE.NE.0)GO TO 52
      IF(NT.EQ.1)GO TO 800
      IE=5
      GO TO 700

C
C      TEST FOR END OF MOD FILE
C
30    IF(MEND.EQ.0)GO TO 40
      NSW=5
      GO TO 500

C
C      TEST FOR CONTINUATION INSERT CARD
C
40    IF(NT.NE.1)GO TO 60
C
C      TEST PREVIOUS MOD CARD AND ERROR CONDITION
C
      IF(IE.NE.0)GO TO 52
      IF(NTL.NE.3)GO TO 800
50    NSW=2
      GO TO 500

C
C      PYPASS CONTINUATION CARDS
C
52    IE=9
      GO TO 700

C
C      GET LINE/CARD NUMBERS
C
60    IE=0
      ENCODE(84,62,NTEM1)MOUB
62    FORMAT(14A6)
      DECODE (80,64,NTEM1)NTEM2
64    FORMAT (8(A6,A4))
      CALL VALUE(NTEM2(3),VN1,IER)

```

```

EDITOK76
EDITOK77
EDITOK78
EDITOK79
EDITOK80
EDITOK81
EDITOK82
EDITOK83
EDITOK84
EDITOK85
EDITOK86
EDITOK87
EDITOK88
EDITOK89
EDITOK90
EDITOK91
EDITOK92
EDITOK93
EDITOK94
EDITOK95
EDITOK96
EDITOK97
EDITOK98
EDITOK99
EDITO100
EDITO101
EDITO102
EDITO103
EDITO104
EDITO105
EDITO106
EDITO107
EDITO108
EDITO109
EDITO110
EDITO111

```

```

        IF(IER.EQ.0) GO TO 68
C
C      SET ERROR FLAG - ILLEGAL LINE ENTRY
C
      66  IE=2
        GO TO 700
      68  N1=VN1
        NSW=3
C
C      TEST FOR FILE POSITION
C
      70  IF(I.LT.N1)GO TO 500
        IF(I.EQ.N1)GO TO 80
C
C      SET ERROR FLAG - MOD OUT OF ORDER
C
        IE=4
        GO TO 700
C
C      TEST TYPE OF CARD
C
      80  GO TO (82,84,110,81),NT
C
C      CONTINUATION INSERT CARD IS ILLEGAL
C
      82  IE=6
        GO TO 700
      81  NSW=6
        GO TO 500
C
C      TEST VALIDITY OF N2
C
      84  CALL VALUE(ITEM2(5),VN2,IER)
        IF (IER.EQ.-1)GO TO 86
        IF (IER.GT.0)GO TO 66
        N2=VN2

```

```

EDIT0112
EDIT0113
EDIT0114
EDIT0115
EDIT0116
EDIT0117
EDIT0118
EDIT0119
EDIT0120
EDIT0121
EDIT0122
EDIT0123
EDIT0124
EDIT0125
EDIT0126
EDIT0127
EDIT0128
EDIT0129
EDIT0130
EDIT0131
EDIT0132
EDIT0133
EDIT0134
EDIT0135
EDIT0136
EDIT0137
EDIT0138
EDIT0139
EDIT0140
EDIT0141
EDIT0142
EDIT0143
EDIT0144
EDIT0145
EDIT0146
EDIT0147

```

	IF(N2.LT.N1)GO TO 90	EDITD148
85	NSW=4	EDITD149
	GO TO 600	EDITD150
86	N2=N1	EDITD151
	GO TO 85	EDITD152
C		EDITD153
C	DELETE LINES OUT OF SEQUENCE	EDITD154
C		EDITD155
90	IE=7	EDITD156
	GO TO 700	EDITD157
C		EDITD158
C	TEST FOR END OF DELETE STRING	EDITD159
C		EDITD160
100	IF (I.LE.N2)GO TO 85	EDITD161
	GO TO 810	EDITD162
C		EDITD163
C	ALTER OPTION	EDITD164
C		EDITD165
110	ENCODE(84,62,NTEM1)INB	EDITD166
	DECODE(80,64,NTEM1)NTEM3	EDITD167
	DO112K=1,16,2	EDITD168
	IF(NTEM3(K).NE.NTEM2(5)) GO TO 112	EDITD169
	IF(NTEM3(K+1).NE.NTEM2(6)) GO TO 112	EDITD170
	GO TO 114	EDITD171
112	CONTINUE	EDITD172
	IE=6	EDITD173
	GO TO 700	EDITD174
114	NTEM3(K)=NTEM2(7)	EDITD175
	NTEM3(K+1)=NTEM2(8)	EDITD176
	ENCODE(80,64,NTEM1)NTEM3	EDITD177
	DECODE(84,62,NTEM1)INB	EDITD178
	GO TO 810	EDITD179
C		EDITD180
C	TEST FOR RANGE ERROR	EDITD181
C		EDITD182
120	IF (OEND.EQ.0)GO TO 70	EDITD183

IE=3	EDIT0184
GO TO 700	EDIT0185
C	EDIT0186
C TEST FOR NAME MATCH	EDIT0187
C	EDIT0188
130 IF(MOD8(5).NE.IN8(2))GO TO 140	EDIT0189
IF(MOD8(6).NE.IN8(3))GO TO 140	EDIT0190
MOD8(4)=1H	EDIT0191
MOD8(5)=1H	EDIT0192
MOD8(6)=1H	EDIT0193
NSW=2	EDIT0194
GO TO 600	EDIT0195
140 IE=1	EDIT0196
GO TO 700	EDIT0197
C	EDIT0198
C WRITE IN RECORD ON NOUT	EDIT0199
C	EDIT0200
500 CALL OUTCD(INB,NOUT)	EDIT0201
C	EDIT0202
C READ IN RECORD - TEST FOR \$ENDOFDECK	EDIT0203
C	EDIT0204
600 I=I+1	EDIT0205
CALL INCD(INB,IN)	EDIT0206
IF(ERFLAG.NE.0)RETURN	EDIT0207
IF(INB(1).EQ.6H\$END 0)OEND=1	EDIT0208
GO TO (130,800,120,100,24,10),NSW	EDIT0209
C	EDIT0210
C PRINT ERROR MESSAGE	EDIT0211
C	EDIT0212
700 WRITE (FERR,702)((IEM(J,IE),J=1,7),MOD8	EDIT0213
702 FORMAT(1H ,7A6,3H - /1X,14A6)	EDIT0214
NEF=1	EDIT0215
GO TO (820,810,810,810,810,820,810,810),IE	EDIT0216
C	EDIT0217
C TEST FOR END OF EDITING	EDIT0218
C	EDIT0219

800	CALL OUTCO(MODB,NOUT)	EDIT0220
810	IF(MEND.EQ.0)GO TO 10	EDIT0221
	IF(CEND.EQ.0)GO TO 10	EDIT0222
820	ERFLAG=NEF	EDIT0223
	RETURN	EDIT0224
	END	EDIT0225
#ELT,I	DORMAN.EDITOR	EDITOR 3
	SUBROUTINE EDITER	EDITOR 5
C		EDITOR 6
C	BUILDS EDIT DECKS	EDITOR 7
C		EDITOR 8
C	PROGRAMMER - S. WRAY	EDITOR 9
C		EDITOR10
	COMMON /MISC/ERFLAG,FERR,KARD(14),ACTION(14)	MISC 2
	INTEGER ERFLAG,FERR,ACTION,PRTFIL	MISC 3
	EQUIVALENCE (PRTFIL,FERR)	MISC 4
	COMMON /FILES/ BASIC,MTAPE,FINAL,S1,S2	FILES 2
	INTEGER BASIC,MTAPE,FINAL,S1,S2	FILES 3
	COMMON /NAMES/ IVER,ONAME(2),MODNAM(2),BNAME(2)	NAMES 2
	INTEGER ONAME,MODNAM,BNAME	NAMES 3
	COMMON /REST/TABLE,USES,FILE,END,DECK,DDECK,BLANK ,BATCH	REST 2
	INTEGER TABLE,USES,FILE,END,DECK,DDECK,BLANK,BATCH	REST 3
	INTEGER BMOD(5),CMOD(6)	EDITOR15
	DIMENSION IU(3),IR(3)	EDITOR16
	DATA IU,IR/0,0,0,2,2,2/	EDITOR17
	WRITE (PRTFIL,5)	EDITOR18
5	FORMAT(21H BEGIN EDIT OPERATION/23H ENTER NAME OF NEW DECK/9H READ	EDITOR19
	*Y -)	EDITOR20
	CALL INCO(KARD,5)	EDITOR21
	DO 10 I=1,14	EDITOR22
10	ACTION(1) = BLANK	EDITOR23
	ACTION(1) = DDECK	EDITOR24
	ACTION(2) = KARD(1)	EDITOR25
	ACTION(3) = KARD(2)	EDITOR26
	ACTION(4) = USES	EDITOR27
	WRITE (PRTFIL,15)	EDITOR28

15	FORMAT(29H ENTER NAME OF REFERENCE DECK/9H READY -)	EDITOR29
	CALL INCD(KARD,5)	EDITOR30
	FINAL = 14	EDITOR31
	ACTION(5) = KARD(1)	EDITOR32
	ACTION(6) = KARD(2)	EDITOR33
	CALL EXTRAN (ACTION(8),ACTION(10))	EDITOR34
	IF((KARD(1).EQ.BNAME(1)).AND.(KARD(2).EQ.BNAME(2))) GO TO 25	EDITOR35
	IF(BNAME(1).EQ.0) GO TO 24	EDITOR36
19	WRITE (PRTFIL,20)	EDITOR37
20	FORMAT(28H REFERENCE DECK IS NOT BASIC/29H DESTRUCT PERMISSION REQ	EDITOR38
	*UIRED/16H ENTER YES OR NO/9H READY -)	EDITOR39
	CALL INCD (KARD,5)	EDITOR40
	IF(KARD(1).EQ.2HNO) RETURN	EDITOR41
	IF(KARD(1).NE.3HYES) GO TO 19	EDITOR42
24	CALL USE(ACTION(5))	EDITOR43
25	REWIND FINAL	EDITOR44
	MODNAM(1) = ACTION(2)	EDITOR45
	MODNAM(2) = ACTION(3)	EDITOR46
	WRITE (PRTFIL,30)	EDITOR47
30	FORMAT(16H ENTER MOD CARDS/25H ENTER DONE WHEN FINISHED/9H READY -	EDITOR48
	*)	EDITOR49
	CALL INCD (KARD,5)	EDITOR50
	S1 = 22	EDITOR51
	S2 = 22	EDITOR52
	REWIND S1	EDITOR53
	IU(1) = S1	EDITOR54
	IU(2) = S1 + 1	EDITOR55
	IU(3) = FINAL	EDITOR56
	CALL INTBUF(IU,1R)	EDITOR57
	CALL OUTCD (ACTION,S1)	EDITOR58
	DO 35 I = 1,14	EDITOR59
35	ACTION(I) = BLANK	EDITOR60
	ACTION(1) = END	EDITOR61
	ACTION(2) = DECK	EDITOR62
	CALL OUTCD(ACTION,S1)	EDITOR63
40	CONTINUE	EDITOR64

```

S1 = S2
REWIND S1
S2 = S1 + 1
IF(S2.GT.23) S2 = 22
REWIND S2
MODBP = 0
MODLP = 5000
MODBQ = 0
MODLQ = 5000
CALL INCD (ACTION,S1)
IF(ERFLAG.NE.0) RETURN
CALL OUTCD (ACTION,S2)
CALL INCD (ACTION,S1)
IF(ERFLAG.NE.0) RETURN
IJ = 1
GO TO 90
50 CONTINUE
IJ = 2
IF(KARD(1).EQ.6HDOONE ) GO TO 500
IF(KARD(1).EQ.6HLIST ) GO TO 55
IF(KARD(1).EQ.6H$DELET ) GO TO 61
IF(KARD(1).EQ.6H$CHANG ) GO TO 62
IF(KARD(1).EQ.6H$INSER ) GO TO 63
IF(KARD(1).EQ.6H$ADJ ) GO TO 63
IF(KARD(1).EQ.6H$ALTER ) GO TO 62
48 CALL OUTCD(KARD,S2)
51 WRITE (PRTFIL,49)
CALL INCD(KARD,5)
GO TO 50
55 CONTINUE
IX = 1
54 DO 540 I=1,5
540 CMOD(I) = KARD(I)
545 ENCODE (30,56,BMOD) (CMOD(I),I=1,5)
56 FORMAT (5A6)
57 FORMAT (3(A6,A4))

```

```

EDITER65
EDITER66
EDITER67
EDITER68
EDITER69
EDITER70
EDITER71
EDITER72
EDITER73
EDITER74
EDITER75
EDITER76
EDITER77
EDITER78
EDITER79
EDITER80
EDITER81
EDITER82
EDITER83
EDITER84
EDITER85
EDITER86
EDITER87
EDITER88
EDITER89
EDITER90
EDITER91
EDITER92
EDITER93
EDITER94
EDITER95
EDITER96
EDITER97
EDITER98
EDITER99
EDITE100

```

```

        DECODE (3J,57,8MOD) CMOD
        CALL VALUE (CMOD(3),V,IER)
        IF (IER.GT.0) GO TO 58
        IST = V
        ISP = 1ST
        IF (CMOD(1).EQ.6H$ALTER ) GO TO 5300
        IF (CMOD(1).EQ.6H$INSER ) GO TO 5300
        CALL VALUE (CMOD(5),V,IER)
        IF (IER.GT.0) GO TO 58
        ISP = V
        IF (ISP.LT.1ST) ISP = 1ST
5300 CONTINUE
        GO TO (53,75,105),IX
        53 CALL LCRO (FINAL,IST,ISP,PRTFIL)
        REWIND FINAL
        CALL INTBUF(IU,IR)
        GO TO 51
        59 FORMAT(36H ILLEGAL NUMERIC ENTRY - REENTER CARD)
        58 WRITE (FERR,59)
        GO TO (51,51,105),IX
        61 IY = 2
        GO TO 64
        62 IY = 1
        GO TO 64
        63 IY = 3
        64 IX = 2
        GO TO 54
        75 INB = IST
        INL = 1SP
        49 FORMAT(9H READY - )
        IF (INB.GE.MODBP) GO TO 80
        IF (INL.GT.MODBP) GO TO 79
        81 CALL OUTCD (ACTION,S2)
        IF (ACTION(1).EQ.END) GO TO 40
        CALL INCD(ACTION,S1)
        GO TO 81

```

```

EDITE101
EDITE102
EDITE103
EDITE104
EDITE105
EDITE106
EDITE107
EDITE108
EDITE109
EDITE110
EDITE111
EDITE112
EDITE113
EDITE114
EDITE115
EDITE116
EDITE117
EDITE118
EDITE119
EDITE120
EDITE121
EDITE122
EDITE123
EDITE124
EDITE125
EDITE126
EDITE127
EDITE128
EDITE129
EDITE130
EDITE131
EDITE132
EDITE133
EDITE134
EDITE135
EDITE136

```

```

79  CONTINUE
    WRITE (FERR,82)
82  FORMAT(40H MOD OVERLAPS EXISTING MOD - RENTER CARD)
    GO TO 51
80  CONTINUE
    IF(INB.GE.MODLP) GO TO 85
    IF(INL.GT.MODLP) GO TO 79
83  MODBP = INB
    MODBQ = INL
    GO TO 48
90  CONTINUE
    IA = 0
    IF(ACTION(1).EQ.6H$DELET ) GO TO 91
    IF(ACTION(1).EQ.6H$CHANG ) GO TO 92
    IF(ACTION(1).EQ.6H$INSER ) GO TO 93
    IF(ACTION(1).EQ.6H$ADD   ) GO TO 93
    IF(ACTION(1).EQ.6H$ALTER ) GO TO 92
    GO TO 106
91  IA = 2
    GO TO 94
92  IA = 1
    GO TO 94
93  IA = 3
94  CONTINUE
    IX = 3
    DO 560 I=1,5
560  CMOD(I) = ACTION(I)
    GO TO 545
105  MODLP = IST
    MODLQ = ISP
106  GO TO (50,120),IJ
85  CONTINUE
    IF(INB.EQ.MODLP) GO TO 160
    IF(ACTION(1).EQ.END) GO TO 83
87  CALL OUTCD(ACTION,S2)
    CALL INCD (ACTION,S1)

```

```

EDITE137
EDITE138
EDITE139
EDITE140
EDITE141
EDITE142
EDITE143
EDITE144
EDITE145
EDITE146
EDITE147
EDITE148
EDITE149
EDITE150
EDITE151
EDITE152
EDITE153
EDITE154
EDITE155
EDITE156
EDITE157
EDITE158
EDITE159
EDITE160
EDITE161
EDITE162
EDITE163
EDITE164
EDITE165
EDITE166
EDITE167
EDITE168
EDITE169
EDITE170
EDITE171
EDITE172

```

```

        IF(ERFLAG.NE.0) RETURN
        IX = 3
        IF(ACTION(1).NE.END) GO TO 90
        MODLP = 5000
        MODLQ = 5000
        GO TO 83
120  IF(IA.EQ.0) GO TO 87
        GO TO 80
160  IF(IA.EQ.1) GO TO 170
        IF(IA.EQ.2) GO TO 180
        IF(IY.EQ.1) GO TO 83
        IF(IY.EQ.3) GO TO 83
        GO TO 200
170  IF(IY.EQ.1) GO TO 83
        GO TO 87
180  IF(IY.EQ.1) GO TO 83
        IF(IY.EQ.3) GO TO 87
        GO TO 200
200  CONTINUE
        WRITE (FERR,210)
210  FORMAT(45H MOD CONFLICTS WITH PRIOR MOD - PLEASE REENTER)
        GO TO 51
500  CONTINUE
510  CALL OUTCD (ACTION,S2)
        IF(ACTION(1).EQ.END) GO TO 520
        CALL INCD (ACTION,S1)
        IF(ERFLAG.NE.0) RETURN
        GO TO 510
520  CONTINUE
        M1 = 13
        REWIND M1
        REWIND S2
        IU(1) = M1
        IU(2) = S2
        CALL INTBUF (IU,IR)
530  CALL INCD (KARD,S2)

```

```

EDITE173
EDITE174
EDITE175
EDITE176
EDITE177
EDITE178
EDITE179
EDITE180
EDITE181
EDITE182
EDITE183
EDITE184
EDITE185
EDITE186
EDITE187
EDITE188
EDITE189
EDITE190
EDITE191
EDITE192
EDITE193
EDITE194
EDITE195
EDITE196
EDITE197
EDITE198
EDITE199
EDITE200
EDITE201
EDITE202
EDITE203
EDITE204
EDITE205
EDITE206
EDITE207
EDITE208

```

```

        IF(ERFLAG.NE.0) RETURN
        CALL OUTCD(KARD,M1)
        IF(KARD(1).NE.END) GO TO 530
        RETURN
        END
#ELT,1 DORMAN.TERM
        SUBROUTINE TERM
C
C          TERMINATES THE RUN
C
C          PROGRAMMER - S. WRAY
C
        COMMON /MISC/ERFLAG,FERR,KARD(14),ACTION(14)
        INTEGER ERFLAG,FERR,ACTION,PRTFIL
        EQUIVALENCE (PRTFIL,FERR)
        IF(ERFLAG.NE.0) WRITE (PRTFIL,10)
10  FORMAT(15H ERROR IS FATAL)
        WRITE (PRTFIL,5)
        5  FORMAT(37H  -- END DORMAN -- RUN TERMINATED --)
        STOP
        END
#ELT,1 DORMAN.A1TA6
        SUBROUTINE A1TA6 (CA84,CA14)
C
C          ENTER WITH 84 WORDS EACH CONTAINING 1 CHARACTER
C          EXIT  WITH 14 WORDS EACH CONTAINING 6 CHARACTERS
C
C          PROGRAMMER: VOIT
C
        DIMENSION CA14(14),CA84(84)
        ENCODE (84,5,CA14) CA84
        5  FORMAT(14(6A1))
        RETURN
        END
#ELT,1 DORMAN.A6TA1
        SUBROUTINE A6TA1 (CA14,CA84)

```

```

EDITE209
EDITE210
EDITE211
EDITE212
EDITE213
TERM      3
TERM      5
TERM      6
TERM      7
TERM      8
TERM      9
TERM     10
MISC      2
MISC      3
MISC      4
TERM     12
TERM     13
TERM     14
TERM     15
TERM     20
TERM     21
A1TA6     3
A1TA6     5
A1TA6     6
A1TA6     7
A1TA6     8
A1TA6     9
A1TA6    10
A1TA6    11
A1TA6    12
A1TA6    13
A1TA6    14
A1TA6    15
A1TA6    16
A6TA1     3
A6TA1     5

```

C		A6TA1	6
C	ENTER WITH 14 WORDS EACH CONTAINING 6 CHARACTERS	A6TA1	7
C	EXIT WITH 84 WORDS EACH CONTAINING 1 CHARACTER	A6TA1	8
C		A6TA1	9
C	PROGRAMMER: VOIT	A6TA1	10
C		A6TA1	11
	DIMENSION CA14(14),CA84(84)	A6TA1	12
	DECODE (84,5,CA14) CA84	A6TA1	13
	5 FORMAT(14(6A1))	A6TA1	14
	RETURN	A6TA1	15
	END	A6TA1	16
	*ELT,I DORMAN.CONV	CONV	3
	SUBROUTINE CONV	CONV	2
C		CONV	6
C	CONVERTS DECKS	CONV	7
C		CONV	8
C	PROGRAMMER - S. WRAY	CONV	9
C		CONV	10
	COMMON /FILES/ BASIC,MTAPE,FINAL,S1,S2	FILES	2
	INTEGER BASIC,MTAPE,FINAL,S1,S2	FILES	3
	COMMON /REST/TABLE,USES,FILE,END,DECK,DDECK,BLANK ,BATCH	REST	2
	INTEGER TABLE,USES,FILE,END,DECK,DDECK,BLANK,BATCH	REST	3
	COMMON /NAMES/ IVER,DNAME(2),MODNAM(2),BNAME(2)	NAMES	2
	INTEGER UNAME,MODNAM,BNAME	NAMES	3
	COMMON /MISC/ERFLAG,FERR,KARD(14),ACTION(14)	MISC	2
	INTEGER ERFLAG,FERR,ACTION,PRTFIL	MISC	3
	EQUIVALENCE (PRTFIL,FERR)	MISC	4
	DIMENSION IU(3),IR(3)	CONV	15
	INTEGER VERS	CONV	16
	DATA VERS/6HVERSION/	CONV	17
	DATA IU,IR/0,0,0,2,2,2/	CONV	18
	WRITE(PRTFIL,5)	CONV	19
	5 FORMAT(29H ENTER NAME OF REFERENCE DECK/9H READY -)	CONV	20
	CALL INCD (KARD,5)	CONV	21
	WRITE (PRTFIL,10)	CONV	22
	10 FORMAT(25H ENTER NAME OF FINAL DECK/9H READY -)	CONV	23

CALL INCD (ACTION,5)	CONV 24
IU(1) = BASIC	CONV 25
CALL INTBUF(IU,IR)	CONV 26
ACTION(3) = KARD(1)	CONV 27
ACTION(4) = KARD (2)	CONV 28
KK = 0	CONV 29
KA = 0	CONV 30
REWIND BASIC	CONV 31
REWIND 22	CONV 32
REWIND 23	CONV 33
C	CONV 34
C ONE OR TWO OR NONE USE PASSES	CONV 35
C	CONV 36
15 CALL INCD (KARD,BASIC)	CONV 37
IF(ERFLAG.NE.0) RETURN	CONV 38
IF(KARD(5).EQ.VERS) GO TO 15	CONV 39
IF(KARD(1).EQ.END) GO TO 40	CONV 40
IF((ACTION(1).EQ.KARD(2)).AND.(ACTION(2).EQ.KARD(3))) KA=KARD(4)	CONV 41
IF((ACTION(3).EQ.KARD(2)).AND.(ACTION(4).EQ.KARD(3))) KK=KARD(4)	CONV 42
GO TO 15	CONV 43
40 CONTINUE	GWT13 2
FINAL = 23	GWT13 3
CALL USE (ACTION(3))	CONV 74
FINAL = 22	GWT13 4
CALL USE (ACTION)	CONV 76
60 CONTINUE	CONV 77
REWIND 22	CONV 78
REWIND 23	CONV 79
REWIND 13	CONV 80
CALL DIFDEC (23,22,13,5)	CONV 81
MODNAM (1) = ACTION (1)	CONV 82
MODNAM (2) = ACTION (2)	CONV 83
RETURN	CONV 84
END	CONV 85
#ELT,1 JORMAN.EXTRAN	EXTRAN 3
SUBROUTINE EXTRAN(ITIME,IDATE)	EXTRAN 5

```

C
C      ENTER WITH 6 CHAR EACH IN ITIM, AND IDTE
C      EXIT WITH 2 WORDS (6 CHAR AND 2 CHAR)
C
C      PROGRAMMER: VOIT
C
      DIMENSION ITIME(2),IDATE(2)
      DIMENSION ITMP(3)
      DATA ISLS /1H//
      DATA INCL/1H./
      CALL EXTRAN (9,ITIM,IDTE)
      DECODE (6,100,ITIM) (ITMP(1),I=1,3)
100  FORMAT (3A2)
      ENCODE (6,110,ITIME) ITMP(1),INCL,ITMP(2),INCL
110  FORMAT (A2,A1,A2,A1)
      ITIME(2)=ITMP(3)
      DECODE (6,100,IDTE) (ITMP(1),I=1,3)
      ENCODE (6,110,IDATE) ITMP(1),ISLS,ITMP(2),ISLS
      IDATE(2) = ITMP(3)
      RETURN
      END
      *ELT,I DORMAN.FILBUF
      SUBROUTINE FILBUF(BEGIN,END,MAX,BUF ,FILE)
C
C      CALLED BY SUBROUTINES
C      JIFDEC,RESTOR,AND SYNCBF
C
C      PROGRAMMER: VOIT
C
      COMMON /MISC/ERFLAG
      COMMON /MISC/FERR
      INTEGER FERR
      INTEGER ERFLAG
      INTEGER BEGIN,END,MAX,BUF(14,20),FILE

```

```

EXTRAN 0
EXTRAN 7
EXTRAN 8
EXTRAN 9
EXTRAN10
EXTRAN11
EXTRAN12
EXTRAN13
EXTRAN14
EXTRAN15
GWT25 3
EXTRAN17
EXTRAN18
EXTRAN19
EXTRAN20
EXTRAN21
EXTRAN22
EXTRAN23
EXTRAN24
EXTRAN25
EXTRAN26
EXTRAN27
FILBUFV3
FILBUFV5
FILBUFV6
FILBUFV7
FILBUFV8
FILBUFV9
FILBUF10
FILBUF11
FILBUF12
FILBUF13
FILBUF14
FILBUF15
FILBUF16
FILBUF17

```

```

      N = BEGIN-1
      IF(N.LT.MAX) GO TO 5
      WRITE (FERR,10) FILE
10    FORMAT(49H FILBUF - INVALID REQUEST TO FILL BUFFER FOR UNIT,I3)
      ERFLAG = 1
      CALL TERM
      5  N = N + 1
      CALL INCD (BUF(1,N),FILE)
      IF(ERFLAG.NE.0) GO TO 16000
      IF (N.GE.MAX) GO TO 15
      IF(BUF(1,N).NE.6H$END 0 ) GO TO 5
15    END = N
      RETURN
16000 CONTINUE
      DO 16001 I=3,14
16001  BUF(I,N)=1H
      BUF(1,N)=6H$END 0
      BUF(2,N)=6HF FILE
      END = N
      RETURN
      END
*ELT,I DORMAN.FNDBUF
      SUBROUTINE FNDBUF (UNIT,X,Y)

C          CHECK TO SEE UNIT IS ACTIVE
C          RETURN MODE IN X
C
C  PROGRAMMER - VOIT
C
      COMMON/MISC/ERFLAG,FERR
      INTEGER ERFLAG,FERR
      COMMON /WORK/ILUNT(3),IIRW(3),IUTBL(3,19),IACT(8,3),NFILES
      INTEGER UNIT,X,Y
      DO 10 I=1,3
      J = I
      IF (UNIT.EQ.IACT(1,I)) GO TO 12

```

```

FILBUF18
FILBUF19
FILBUF20
FILBUF21
FILBUF22
FILBUF23
FILBUF24
FILBUF25
FILBUF26
FILBUF27
FILBUF28
FILBUF29
FILBUF30
FILBUF31
FILBUF32
FILBUF33
FILBUF34
FILBUF35
FILBUF36
FILBUF37
FILBUF38
FNDBUF 3
FNDBUF 5
FNDBUF 6
FNDBUF 7
FNDBUF 8
FNDBUF 9
FNDBUF10
FNDBUF11
FNDBUF12
FNDBUF13
WORK 2
FNDBUF15
FNDBUF16
FNDBUF17
FNDBUF18

```

```

10    CONTINUE
      WRITE (FERR,1000)  UNIT
1000  FORMAT(28H FNDBUF - FATAL ERROR - UNIT,13,11H NOT ACTIVE)
      ERFLAG = 1
      CALL TERM
12    CONTINUE
      Y = J
      X = IACT(2,J)
      RETURN
      END
#ELT,I DORMAN.GETGEN
      SUBROUTINE GETGEN(NAME,IFILE,LIST,NLIST)
C
C  GET GENEALOGY
C  PROGRAMMER - B. GOLD
C
      DIMENSION NAME(2),LIST(2,20)
      COMMON /MISC/ERFLAG,FERR,KARD(14),ACTION(14)
      INTEGER ERFLAG,FERR,ACTION,PRTFIL
      EQUIVALENCE (PRTFIL,FERR)
      COMMON /REST/TABLE,USES,FILE,END,DECK,DDECK,BLANK ,BATCH
      INTEGER TABLE,USES,FILE,END,DECK,DDECK,BLANK,BATCH
      DIMENSION IU(3),IR(3)
      DATA IU/2,2,2/,IU/0,0,0/
      IU(1) = IFILE
      CALL INTBUF(IU,IR)
      REWIND IFILE
      ERFLAG = 0
      LIST(1,1) = NAME(1)
      LIST(2,1) = NAME(2)
      NLIST = 1
10    CALL INCD(KARD,IFILE)
      IF(ERFLAG.NE.0) RETURN
      IF(KARD(1).NE.END ) GO TO 30
      IF(KARD(2).NE.TABLE) GO TO 30
      WRITE (FERR,20) NAME, ((LIST(I,J), I=1,2), J=1,NLIST)

```

```

FNDBUF19
FNDBUF20
FNDBUF21
FNDBUF22
FNDBUF23
FNDBUF24
FNDBUF25
FNDBUF26
FNDBUF27
FNDBUF28
GETGEN 3
GETGEN 5
GETGEN 6
GETGEN 7
GETGEN 8
GETGEN 9
GETGEN10
MISC 2
MISC 3
MISC 4
REST 2
REST 3
GETGEN13
GETGEN14
GETGEN15
GETGEN16
GETGEN17
GETGEN18
GETGEN19
GETGEN20
GETGEN21
GETGEN22
GETGEN23
GETGEN24
GETGEN25
GETGEN26

```

20	FORMAT(36H0GETGEN ERROR IN GENEALOGY FOR DECK 2A6/44H COULD NOT FIGETGEN27	
	*ND LAST DECK IN FOLLOWING LIST-//(5X2A6))	GETGEN28
	ERFLAG = 1	GETGEN29
	RETURN	GETGEN30
30	IF (KARD(2).NE.LIST(1,NLIST)) GO TO 10	GETGEN31
	IF (KARD(3).NE.LIST(2,NLIST)) GO TO 10	GETGEN32
	IF(KARD(4).EQ.BLANK) RETURN	GETGEN33
	NLIST = NLIST+1	GETGEN34
	LIST(1,NLIST) = KARD(5)	GETGEN35
	LIST(2,NLIST) = KARD(6)	GETGEN36
	GO TO 10	GETGEN37
	END	GETGEN38
#ELT,I	DOORMAN.INCRT	INCR 3
	SUBROUTINE INCRT	INCR 5
C		INCR 6
C	CREATES A DATA BASE	INCR 7
C		INCR 8
C	PROGRAMMER - S. WRAY	INCR 9
C		INCR 10
	DIMENSION IU(3),IR(3)	INCR 11
	COMMON /FILES/ BASIC,MTAPE,FINAL,S1,S2	FILES 2
	INTEGER BASIC,MTAPE,FINAL,S1,S2	FILES 3
	COMMON /REST/TABLE,USES,FILE,END,DECK,DDECK,BLANK ,BATCH	REST 2
	INTEGER TABLE,USES,FILE,END,DECK,DDECK,BLANK,BATCH	REST 3
	COMMON /MISC/ERFLAG,FERR,KARD(14),ACTION(14)	MISC 2
	INTEGER ERFLAG,FERR,ACTION,PRTFIL	MISC 3
	EQUIVALENCE (PRTFIL,FERR)	MISC 4
	COMMON /VCARD/ILBL(14)	INCR 15
	DATA IR/2,2,2/	INCR 16
	IU(1) = 5	INCR 17
	IU(2) = BASIC	INCR 18
	IU(3) = 0	INCR 19
	CALL INTBUF(IU,IR)	INCR 20
	REWIND BASIC	INCR 21
	CALL EXTRAN (ILBL(12),ILBL(10))	INCR 22
	CALL OUTCD (ILBL,BASIC)	INCR 23

```

DO 5 I = 1,14
ACTION(I) = BLANK
5 KARD(I) = BLANK
KARD(1) = ENG
KARD(2) = TABLE
KARD(3) = 1HE
ACTION(1) = DDECK
ACTION(2) = 6HTESTOR
ACTION(4) = USES
ACTION(5) = 5H TEST
CALL OUTCD (ACTION,BASIC)
ACTION(4) = BLANK
ACTION(5) = BLANK
ACTION(2) = 5H TEST
CALL OUTCD (ACTION,BASIC)
CALL OUTCD(KARD,BASIC)
ACTION(2) = 6HTESTOR
ACTION(4) = USES
ACTION(5) = 5H TEST
CALL OUTCD(ACTION,BASIC)
KARD(2) = DECK
KARD(3) = BLANK
CALL OUTCD (KARD,BASIC)
ACTION(2) = 5H TEST
ACTION(4) = BLANK
ACTION(5) = BLANK
CALL OUTCD (ACTION,BASIC)
CALL OUTCD (KARD,BASIC)
KARD(2) = FILE
CALL OUTCD (KARD,BASIC)
REWIND 21
CALL ADDX(21)
RETURN
END
*ELT,I DORMAN.INCD
SUBROUTINE INCD (ICRD,IUNIT)

```

```

INCRT 24
INCRT 25
INCRT 26
INCRT 27
INCRT 28
INCRT 29
INCRT 30
INCRT 31
INCRT 32
INCRT 33
INCRT 34
INCRT 35
INCRT 36
INCRT 37
INCRT 38
INCRT 39
INCRT 40
INCRT 41
INCRT 42
INCRT 43
INCRT 44
INCRT 45
INCRT 46
INCRT 47
INCRT 48
INCRT 49
INCRT 50
INCRT 51
INCRT 52
INCRT 53
INCRT 54
INCRT 55
INCRT 56
INCRT 57
INCDM 3
INCDM 5

```

C		INCDM 6
C	PUT IN FORCE OF +ASG	INCDM 7
C		INCDM 8
C	MODE 1 - READ 55 CARDS	INCDM 9
C	MODE 2 - READ 1 CARD	INCDM 10
C	MODE 3 - READ 1 CARD, MAY BE PACKED	INCDM 11
C		INCDM 12
C	PROGRAMMER: VOIT	INCDM 13
C		INCDM 14
	INTEGER ICRO(14)	INCDM 15
	COMMON /REST/TABLE,USES,FILE,END,DECK,DDECK,BLANK ,BATCH	REST 2
	INTEGER TABLE,USES,FILE,END,DECK,DDECK,BLANK,BATCH	REST 3
	COMMON /WORK/IUNIT(3),IIRW(3),IUTBL(3,19),IACT(8,3),NFILES	WORK 2
	COMMON /BFRS/ XXX(54)	BFRS 2
	COMMON /BFRS/ IMOD1(14,55,2),IWK1(168,2),ITEMP1(84),ITEMP2(84)	BFRS 3
	COMMON /MISC/ERFLAG,FERR,KARD(14),ACTION(14)	MISC 2
	INTEGER ERFLAG,FERR,ACTION,PRTFIL	MISC 3
	EQUIVALENCE (PRTFIL,FERR)	MISC 4
	DIMENSION ITMP1(1),ITMP2(1)	INCDM 20
	EQUIVALENCE (ITMP1(1),ITEMP1(1)),(ITEMP2(1),ITMP2(1))	INCDM 21
	IF(IUNIT.EQ.5) GO TO 5	INCDM 22
	CALL FND8UF (IUNIT,IX,IY)	INCDM 23
	IF (IX.NE.2) GO TO 100	INCDM 24
C		INCDM 25
C	MODE EQ 2 -- READ ONE CARD	INCDM 26
C		INCDM 27
	IF (IACT(4,IY).EQ.1) GO TO 510	INCDM 28
	CALL UREAD (IUNIT,ICRO)	INCDM 29
C		INCDM 30
C	IF END OF FILE, SET FLAG	INCDM 31
C		INCDM 32
	20 CONTINUE	INCDM 33
	IF(ICRO(1).NE.END) RETURN	INCDM 34
	IF(ICRO(2).NE.FILE) RETURN	INCDM 35
	IACT(4,IY) = 1	INCDM 36
	IF(IUNIT.EQ.5) CALL TERM	INCDM 37

```

      RETURN
100  CONTINUE
      IF (IX.NE.1) GO TO 200
C
C      MODE 1
C      IS BUFFER EMPTY
C
      IXX = IACT(8,IY)
      IYY = IACT(7,IY)
      IF (IACT(4,IY).EQ.1) GO TO 510
      GO TO 155
C
C      CAN WORK AREA TAKE 84 CHARACTERS
C
117  CONTINUE
      IF (IXX.GT.84) GO TO 170
      IF (IACT(4,IY).EQ.1) GO TO 170
C
C      PUT 84 CHAR FROM BUFFER INTO TOP OF WORK AREA
C
      CALL A6TA1(IMOD1(1,IYY,IY),IWK1(IXX+1,IY))
      IXX = IXX + 84
      IYY = IYY + 1
      IF(IYY.NE.56) GO TO 117
C
C      READ IN NEXT BUFFER FULL
C
155  CONTINUE
      DO 160 I=1,55
      READ (IUNIT,156,END=510,ERR=510) (IMOD1(J,1,IY),J=1,770)
156  FORMAT(770A6)
160  CONTINUE
      IYY = 1
      IACT(4,IY) = 0
      GO TO 117
C

```

```

INCDM 38
INCDM 39
INCDM 40
INCDM 41
INCDM 42
INCDM 43
INCDM 44
INCDM 45
INCDM 46
INCDM 47
INCDM 48
INCDM 49
INCDM 50
INCDM 51
INCDM 52
INCDM 53
INCDM 54
INCDM 55
INCDM 56
INCDM 57
INCDM 58
INCDM 59
INCDM 60
INCDM 61
INCDM 62
INCDM 63
INCDM 64
INCDM 65
INCDM 66
INCDM 71
INCDM 72
INCDM 74
INCDM 75
INCDM 76
INCDM 77
INCDM 78

```



```

C          MOVE CARDS FROM WORK AREA INTO CARD
C
170  CONTINUE
    CALL UNPAC (ITEMP1,ITEMP2,ICH)
    CALL A1TA6(ITEMP2,ICRD)
C
C          MOVE DOWN WORK AREA
C
    K = IXX-ICH
    IF(K.EQ.0) GO TO 195
    DO 190 I=1,K
    ICH1 = 1+ICH
    IWK1(I,IY) = IWK1(ICH1,IY)
190  CONTINUE
195  CONTINUE
    IXX = K
    IACT(8,IY)=IXX
    IACT(7,IY)=IYY
    GO TO 20
200  CONTINUE
    IF(IX.NE.3) GO TO 500
C
C          MODE 3
C
5  CONTINUE
    CALL UREAD (IUNIT,ICRD)
    CALL A6TA1(ICRD,ITEMP1)
    CALL UNPAC (ITEMP1,ITEMP2,ICH)
    CALL A1TA6(ITEMP2,ICRD)
    IF(BATCH.EQ.IUNIT) CALL OUTCD(ICRD,PRTFIL)
    GO TO 20
C
C          ERROR EXIT
C
500  CONTINUE
    WRITE (FERR,1000) IUNIT,IX

```

```

INCDM 79
INCDM 80
INCDM 81
INCDM 82
INCDM 83
INCDM 84
INCDM 85
INCDM 86
INCDM 87
INCDM 88
INCDM 89
INCDM 90
INCDM 91
INCDM 92
INCDM 93
INCDM 94
INCDM 95
INCDM 96
INCDM 97
INCDM 98
INCDM 99
INCDM100
INCDM101
INCDM102
INCDM103
INCDM104
INCDM105
INCDM106
INCDM107
INCDM108
INCDM109
INCDM110
INCDM111
INCDM112
INCDM113
INCDM114

```

1000	FORMAT(26H INCD - MODE ERROR -- UNIT,I3,5H MODE,I3)	INCDM115
	ERFLAG = 1	INCDM116
	CALL TERM	INCDM117
510	CONTINUE	INCDM118
	WRITE (FERR,2000) IUNIT	INCDM119
2000	FORMAT(27H INCD - END OF FILE ON UNIT,I3)	INCDM120
	ERFLAG = 1	INCDM121
	IF(IUNIT.EQ.5) CALL TERM	INCDM122
	RETURN	INCDM123
	END	INCDM124
#ELT,I	DORMAN.INSERT	INSERT 3
	SUBROUTINE INSERT(NSTP,II)	INSERT 5
C		INSERT 6
C	INSERT	INSERT 7
C		INSERT 8
	COMMON /BFRS/ ISYN(2,20),ISYN1,IT1,IT2,IT3,FULL,CN1,CN2,LIN1,LIN2	BFRS1 2
	* ,NWBUF,NB1,NB2,ICN1,ICN2	BFRS1 3
	COMMON /BFRS/ BUF1(14,50),BUF2(14,50),AAA(700)	BFRS1 4
	INTEGER CN1,CN2,BUF1,BUF2	BFRS1 5
	LOGICAL FULL	BFRS1 6
	INTEGER FIN1,FIN2	INSERT10
	EQUIVALENCE (FOUT,IT3)	INSERT11
	EQUIVALENCE (FIN2,IT2)	INSERT12
	EQUIVALENCE (FIN1,IT1)	INSERT13
	INTEGER FOUT	INSERT14
	IF(II.NE.0) GO TO 10	INSERT15
	J = LIN1 - 2	INSERT16
	WRITE(FOUT,46) J	INSERT17
46	FORMAT (7H\$INSERT,I13)	INSERT18
10	CONTINUE	INSERT19
50	CONTINUE	INSERT20
	LSTCD = (NB2-CN2) + LIN2	INSERT21
	IF (NSTP.GT.LSTCD) GO TO 100	INSERT22
	KK = NSTP-LIN2+CN2-1	INSERT23
	WRITE (FOUT,9) ((BUF2(I,K),I=1,14),K=CN2,KK)	INSERT24
	KK = KK - CN2 + 1	INSERT25

```

        LIN2 = LIN2 + KK
        CN2 = CN2 + KK
        RETURN
100  CONTINUE
        KK = NB2 - CN2 + 1
        WRITE (FOUT,9) ((BUF2(I,K),I=1,14),K=CN2,NB2)
        LIN2 = LIN2+KK
        CALL FILBUF (1,NB2,NWBUF,BUF2,FIN2)
        CN2 = 1
        GO TO 50
9     FORMAT (14A6)
        END
*ELT,1 DORMAN.INTBUF
        SUBROUTINE INTBUF (IUNT,IRW)
C
C         ENTER WITH 3 UNITS AND READ/WRITE STATUS
C         INITIALIZE ACTIVE TABLE
C
C         PROGRAMMER: VOIT
C
        COMMON /MISC/ERFLAG,FERR,KARD(14),ACTION(14)
        INTEGER ERFLAG,FERR,ACTION,PRTFIL
        EQUIVALENCE (PRTFIL,FERR)
        COMMON /WORK/IUNT(3),IIRW(3),IUTBL(3,19),IACT(8,3),NFILES
        COMMON /BFRS/ XXX(54)
        COMMON /BFRS/ IMOD1(14,55,2),IWK1(168,2),ITEMP1(84),ITEMP2(84)
        DIMENSION IUNT(3),IRW(3)
C
C         IUTBL --- UNIT TABLE
C         1 - UNIT NUMBER
C         2 - MODE
C
C         IACT --- 3 ACTIVE FILES
C         1 - UNIT NUMBER
C         2 - MODE
C         3 - 0=READ ONLY, 1=WRITE ONLY, 2=READ OR WRITE

```

```

INSERT26
INSERT27
INSERT29
INSERT30
INSERT31
INSERT32
INSERT33
INSERT34
INSERT35
INSERT36
INSERT37
INSERT38
INTBUF 3
INTBUF 5
INTBUF 6
INTBUF 7
INTBUF 8
INTBUF 9
INTBUF10
INTBUF11
MISC 2
MISC 3
MISC 4
WORK 2
BFRS 2
BFRS 3
INTBUF15
INTBUF16
INTBUF17
INTBUF18
INTBUF19
INTBUF20
INTBUF21
INTBUF22
INTBUF23
INTBUF24

```

```

C
C      3 - READ=0,WRITE=1
C      4 - FILE STATUS 1=END OF FILE, 2=REWIND,0=IN USE
C      5 - BUFFER INDEX
C      6 - WORK AREA INDEX
C      7 - BUFFER CARD COUNTER
C      8 - WORK AREA CHARACTER COUNTER
C

```

```

      DIMENSION IMOD2(1780)
      EQUIVALENCE (IMOD2(1),IMOD1(1,1,1))
      KK = NFILES
      DO100 L=1,3
      DO 10 I=1,KK
      J = I
      IF(IUNT(L).EQ.0) GO TO 11
      IF (IUNT(L).EQ.IUTBL(1,I)) GO TO 12
10    CONTINUE
      GO TO 260
11    DO 9 K = 1,6
      9    IACT(K,L) = 0
      GO TO 100

```

```

C
C      UNIT FOUND
C
12    CONTINUE
      IACT(1,L) = IUNT(L)
      IACT(2,L) = IUTBL(2,J)
      IACT(3,L) = IRW(L)
      K = IACT(1,L)
      IACT(4,L) = 2
100  CONTINUE

```

```

C
C      ASSIGN BUFFER FOR MODE 1
C
      K = 0
      DO 200 L=1,3

```

```

INTBUF25
INTBUF26
INTBUF27
INTBUF28
INTBUF29
INTBUF30
INTBUF31
INTBUF32
INTBUF33
INTBUF34
INTBUF35
INTBUF36
INTBUF37
INTBUF38
INTBUF39
INTBUF40
INTBUF41
INTBUF42
INTBUF43
INTBUF44
INTBUF45
INTBUF46
INTBUF47
INTBUF48
INTBUF49
INTBUF50
INTBUF51
INTBUF52
INTBUF53
INTBUF54
INTBUF55
INTBUF56
INTBUF57
INTBUF58
INTBUF59
INTBUF60

```

```

        IF (IACT(2,L).NE.1) GO TO 150
        K = K+1
        IF (K.GT.2) GO TO 250
        DO 5 I=1,1780
        IMOD2(1)=0
5      CONTINUE
        IACT(5,L) = K
        IACT(6,L) = K
        GO TO 160
150    IACT(5,L) = 0
C
C      FORCE REFERENCE TO +ASG
C
        IACT(6,L) = 0
        IACT(7,L) = 0
        IACT(8,L) = 0
        GO TO 200
160    CONTINUE
        IACT(7,L) = 1
        IACT(8,L) = 0
200    CONTINUE
        RETURN
C
C      ERROR EXIT
C
260    CONTINUE
        WRITE (FERR,1000) IUNT(L)
1000   FORMAT(14H INTBUF - UNIT,I3,24H NOT FOUND IN UNIT TABLE)
1001   ERFLAG = 1
        CALL TERM
250    CONTINUE
        WRITE (FERR,1010) IUNT(L),(IACT(J,L),J=1,8)
1010   FORMAT(29H INTBUF - CANNOT ASSIGN UNITS,I3/5X,8I10)
        GO TO 1001
        END
*ELT,I DORMAN.LABLER

```

```

INTBUF61
INTBUF62
INTBUF63
INTBUF64
INTBUF65
INTBUF66
INTBUF67
INTBUF68
INTBUF69
INTBUF70
INTBUF71
INTBUF72
INTBUF73
INTBUF74
INTBUF75
INTBUF76
INTBUF77
INTBUF78
INTBUF79
INTBUF80
INTBUF81
INTBUF82
INTBUF83
INTBUF84
INTBUF85
INTBUF86
INTBUF87
INTBUF88
INTBUF89
INTBUF90
INTBUF91
INTBUF92
INTBUF93
INTBUF94
INTBUF95
LABLER 3

```

C	SUBROUTINE LABLER (IVER,FILE1,FILE2)	LABLER 5
C	WRITE LABEL AND VERSION ON FILE2	LABLER 6
C	THEN COPY FILE1 TO FILE2	LABLER 7
C		LABLER 8
C	PROGRAMMER: VOIT	LABLER 9
C		LABLER10
	COMMON /MISC/ERFLAG,FERR	LABLER11
	INTEGER ERFLAG,FERR	LABLER12
	DIMENSION IFL(3),IRW(3)	LABLER13
	DIMENSION ICRD(14)	LABLER14
	COMMON /VCARD/ILBL(14)	LABLER15
	INTEGER FILE1,FILE2	LABLER16
C		LABLER17
C	FILL IN VERSION NUMBER,DATE AND TIME IN ILBL	LABLER18
C		LABLER19
	ILBL(7) = IVER	LABLER20
	CALL EXTRAN (ILBL(9),ILBL(11))	LABLER21
C		LABLER22
C	INITIALIZE FILES AND TRANSFER DATA	LABLER23
C		LABLER24
	IFL(1) = FILE1	LABLER25
	IFL(2) = FILE2	LABLER26
	IFL(3) = 0	LABLER27
	IRW(1) = 0	LABLER28
	IRW(2) = 1	LABLER29
	CALL INIBUF (IFL,IRW)	LABLER30
	CALL OUTCD (ILBL,FILE2)	LABLER31
	CALL INCD (ICRD,FILE1)	LABLER32
	IF(ERFLAG.NE.0) GO TO 50	LABLER33
30	CONTINUE	LABLER34
	CALL INCD (ICRD,FILE1)	LABLER35
	IF(ERFLAG.NE.0) GO TO 50	LABLER36
	CALL OUTCD (ICRD,FILE2)	LABLER37
	IF(ICRD(1).NE.6H\$END 0) GO TO 30	LABLER38
	IF(ICRD(2).NE.6HF FILE) GO TO 30	LABLER39
		LABLER40

```

      RETURN
50 WRITE(FERR,51)
51 FORMAT(41H LABEL - NEW DATA FILE COULD NOT BE SAVED )
      ERFLAG = 1
      RETURN
      END
*ELT,I DORMAN.LCRD
      SUBROUTINE LCRD(FILE,ISTRT,ISTP,PRTFIL)
C
C      LIST CARDS ON PRTFIL
C
C      PROGRAMMER: VOIT
C
      INTEGER PRTFIL,FILE,CARD(14)
      IF(PRTFIL.NE.6) WRITE (PRTFIL,6)
6  FORMAT(1H1)
      WRITE (PRTFIL,5)
5  FORMAT(1H3)
      IF(ISTRT.LT.0) ISTRT=0
      IF (ISTP.LT.1STRT) GO TO 50
10  CONTINUE
      IF(ISTRT.EQ.0) GO TO 25
      DO 20 I=1,ISTRT
      CALL INCD (CARD,FILE)
      IF(CARD(1).NE.6H$END 0 ) GO TO 20
      IF(CARD(2).EQ.6HF FILE ) GO TO 100
20  CONTINUE
25  CONTINUE
      DO 30 I=ISTRT,ISTP
      CALL INCD(CARD,FILE)
      DO 27 L=1,14
      J = 15 - L
      IF(CARD(J).NE.1H ) GO TO 28
27  CONTINUE
28  CONTINUE
      WRITE (PRTFIL,26) I,(CARD(LL),LL=1,J)

```

```

LABLER+1
LABLER+2
LABLER+3
LABLER+4
LABLER+5
LABLER+6
LCRD 3
LCRD 5
LCRD 6
LCRD 7
LCRD 8
LCRD 9
LCRD 10
LCRD 11
LCRD 12
LCRD 13
LCRD 14
LCRD 15
LCRD 16
LCRD 17
LCRD 18
LCRD 19
LCRD 20
LCRD 21
LCRD 22
LCRD 23
LCRD 24
LCRD 25
LCRD 26
LCRD 27
LCRD 28
LCRD 29
LCRD 30
LCRD 31
LCRD 32
LCRD 33

```

26	FORMAT(15,1X,14A6)	LORD	34
	IF(CARD(1).NE.6HEND O) GO TO 24	LORD	35
	IF(CARD(2).EQ.6HF FILE) GO TO 100	LORD	36
24	CONTINUE	LORD	37
30	CONTINUE	LORD	38
	WRITE (PRTFIL,5)	LORD	39
	RETURN	LORD	40
50	ISTP = ISTRT	LORD	41
	GO TO 10	LORD	42
100	CONTINUE	LORD	43
	IF(PRTFIL.NE.6) RETURN	LORD	44
	WRITE (PRTFIL,1000) I	LORD	45
1000	FORMAT(45H LORD - COUNT TOO HIGH, FOUND END OF FILE AT ,15)	LORD	46
	WRITE (PRTFIL,5)	LORD	47
	RETURN	LORD	48
	END	LORD	49
7ELT,I	DORMAN.LISTER	LISTER	3
	SUBROUTINE LISTER	LISTER	5
C		LISTER	6
C	LIST OPTIONS SUBROUTINE	LISTER	7
C		LISTER	8
C	PROGRAMMER - S. WRAY	LISTER	9
C		LISTER	10
	DIMENSION IU(3),IR(3)	LISTER	11
	COMMON /NAMES/ IVER,DNAME(2),MODNAM(2),BNAME(2)	NAMES	2
	INTEGER DNAME,MODNAM,BNAME	NAMES	3
	COMMON /FILES/ BASIC,MTAPE,FINAL,S1,S2	FILES	2
	INTEGER BASIC,MTAPE,FINAL,S1,S2	FILES	3
	COMMON /MISC/ERFLAG,FERR,KARD(14),ACTION(14)	MISC	2
	INTEGER ERFLAG,FERR,ACTION,PRTFIL	MISC	3
	EQUIVALENCE (PRTFIL,FERR)	MISC	4
	WRITE (PRTFIL,66)	LISTER	15
66	FORMAT(63H ENTER LIST OPTION (CONTENTS, GENEALOGY, COUNT, CARDS OR	LISTER	16
	* PRINT) /9H READY -)	LISTER	17
	CALL INCO (KARD,5)	LISTER	18
	IF(KARD(1).EQ.6HCONTEN) GO TO 69	LISTER	19

IF(KARD(1).EQ.6HGENEAL) GO TO 67	LISTER20
IF(KARD(1).EQ.6HCOUNT) GO TO 70	LISTER21
IF(KARD(1).EQ.6HCARDS) GO TO 80	LISTER22
IF(KARD(1).EQ.6HPRINT) GO TO 90	LISTER23
RETURN	LISTER24
67 WRITE (PRTFIL,68)	LISTER25
68 FORMAT(27H ENTER NAME OF DECK DESIRED/9H READY -)	LISTER26
CALL INCD(ACTION,5)	LISTER27
CALL LISTG(ACTION,BASIC)	LISTER28
RETURN	LISTER29
69 CALL LISTTC(BASIC)	LISTER30
100 RETURN	LISTER31
70 CONTINUE	LISTER32
IX = 1	LISTER33
GO TO 75	LISTER34
80 CONTINUE	LISTER35
IX = 2	LISTER36
GO TO 75	LISTER37
90 CONTINUE	LISTER38
IX = 3	LISTER39
75 CONTINUE	LISTER40
71 FORMAT(19H ENTER NAME OF DECK/18H OR FILE NUMBER/24H OR	CURRLISTER41
*ENT FILE NAME/11H OR DONE/9H READY -)	LISTER42
WRITE (PRTFIL,71)	LISTER43
CALL INCD (KARD,5)	LISTER44
IF(KARD(1).EQ.4HDONE) RETURN	LISTER45
ITAPE=0	LISTER46
CALL VALUE (KARD,V,IERR)	LISTER47
IF(IERR.EQ.0) ITAPE = V	LISTER48
IF(KARD(1).EQ.6HBASIC) ITAPE = BASIC	LISTER49
IF(KARD(1).EQ.6HFINAL) ITAPE = FINAL	LISTER50
IF((KARD(1).EQ.BNAME(1)).AND.(KARD(2).EQ.BNAME(2))) ITAPE = FINAL	LISTER51
FINAL = 14	LISTER52
IF(ITAPE.EQ.0) CALL USE (KARD)	LISTER53
IF(ERFLAG.NE.0) RETURN	LISTER54
IF(ITAPE.EQ.0) ITAPE=FINAL	LISTER55

```

      IU(1) = 5
      IU(2) = ITAPE
      IU(3) = 0
      IR(1) = 2
      IR(2) = 2
      IR(3) = 2
      REWIND ITAPE
      CALL INTBUF(IU,IR)
      IF(IX.EQ.1) GO TO 86
      IF(IX.EQ.3) GO TO 95
81  WRITE (PRTFIL,82)
82  FORMAT(27H ENTER NUMBER OF FIRST CARD/9H READY - )
      CALL INCD (KARD,5)
      CALL VALUE (KARD,V,IERR)
      IF (IERR.NE.0) GO TO 81
      ISTART = V
83  WRITE (PRTFIL,84)
84  FORMAT(27H ENTER NUMBER OF LAST CARD /9H READY - )
      CALL INCD (KARD,5)
      ISTOP = 0
      IF(KARD(1).EQ.1H ) ISTOP = ISTART
      IF(ISTOP.NE.0) GO TO 85
      CALL VALUE (KARD,V,IERR)
      IF(IERR.NE.0) GO TO 83
      ISTOP = V
85  CONTINUE
      CALL LCRD (ITAPE,ISTART,ISTOP,6)
      GO TO 80
86  CALL COUNT(ITAPE)
      GO TO 79
95  CONTINUE
      CALL LCRD(ITAPE,0,50000,20)
      GO TO 90
      END
      *ELT,I DORMAN.LISTG
      SUBROUTINE LISTG(NAME,IFILE)

```

```

      LISTER56
      LISTER57
      LISTER58
      LISTER59
      LISTER60
      LISTER61
      LISTER62
      LISTER63
      LISTER64
      LISTER65
      LISTER66
      LISTER67
      LISTER68
      LISTER69
      LISTER70
      LISTER71
      LISTER72
      LISTER73
      LISTER74
      LISTER75
      LISTER76
      LISTER77
      LISTER78
      LISTER79
      LISTER80
      LISTER81
      LISTER82
      LISTER83
      LISTER84
      LISTER85
      LISTER86
      LISTER87
      LISTER88
      LISTER89
      LISTG 3
      LISTG 5

```

C		LISTG 6
C	LIST GENEALOGY	LISTG 7
C	PROGRAMMER - B. GOLD	LISTG 8
C		LISTG 9
	DIMENSION LIST(2,20), NAME(2)	LISTG 10
	COMMON /MISC/ERFLAG,FERR,KARD(14),ACTION(14)	MISC 2
	INTEGER ERFLAG,FERR,ACTION,PRTFIL	MISC 3
	EQUIVALENCE (PRTFIL,FERR)	MISC 4
C		LISTG 12
	CALL GETGEN(NAME,IFILE,LIST,NLIST)	LISTG 13
	IF (ERFLAG.NE.0) RETURN	LISTG 14
	WRITE (PRTFIL,10) NAME	LISTG 15
10	FORMAT(20H0GENEALOGY FOR DECK 2A6)	LISTG 16
	IF (NLIST.EQ.1) GO TO 40	LISTG 17
	DO 20 N = 2,NLIST	LISTG 16
20	WRITE (PRTFIL,30) LIST(1,N-1),LIST(2,N-1), LIST(1,N),LIST(2,N)	LISTG 19
30	FORMAT(7H \$DECK 2A6,6H USES ,2A6)	LISTG 20
40	WRITE (PRTFIL,50) LIST(1,NLIST), LIST(2,NLIST)	LISTG 21
50	FORMAT(7H \$DECK 2A6)	LISTG 22
	WRITE (PRTFIL,60)	LISTG 23
60	FORMAT(1H0)	LISTG 24
	RETURN	LISTG 25
	END	LISTG 26
*ELT,I	DORMAN.LISTTC	LISTTC 3
	SUBROUTINE LISTTC(IFILE)	LISTTC 5
C		LISTTC 6
C	LIST TABLE OF CONTENTS	LISTTC 7
C		LISTTC 8
C	PROGRAMMER - B. GOLD	LISTTC 9
C		LISTTC 10
	DIMENSION IU(3),IR(3)	LISTTC 11
	COMMON /REST/TABLE,USES,FILE,END,DECK,DDECK,BLANK ,BATCH	REST 2
	INTEGER TABLE,USES,FILE,END,DECK,DDECK,BLANK,BATCH	REST 3
	COMMON /MISC/ERFLAG,FERR,KARD(14),ACTION(14)	MISC 2
	INTEGER ERFLAG,FERR,ACTION,PRTFIL	MISC 3
	EQUIVALENCE (PRTFIL,FERR)	MISC 4

INTEGER VERS	LISTTC14
DATA VERS/6HVERSIO/	LISTTC15
DATA IR/2,2,2/,IU/0,0,0/	LISTTC16
IU(1) = IFILE	LISTTC17
CALL INTBUF(IU,IR)	LISTTC18
REWIND IFILE	LISTTC19
ERFLAG = 0	LISTTC20
WRITE (PRTFIL,10)	LISTTC21
10 FORMAT(21H0 TABLE OF CONTENTS/)	LISTTC22
20 CALL INCD(KARD,IFILE)	LISTTC23
IF(ERFLAG.NE.0) RETURN	LISTTC24
IF(KARD(5).EQ.VERS) GO TO 25	LISTTC25
IF((KARD(1).NE.DDECK).AND.(KARD(1).NE.END)) GO TO 60	LISTTC26
25 CONTINUE	LISTTC27
CALL OUTCD (KARD,PRTFIL)	LISTTC28
IF(KARD(1).NE.END) GO TO 20	LISTTC29
IF(KARD(2).NE.TABLE) GO TO 20	LISTTC30
RETURN	LISTTC31
60 ERFLAG = 1	LISTTC32
WRITE (FERR,70) KARD	LISTTC33
70 FORMAT(49H0LISTC - ERRONEOUS CARD WITHIN TABLE OF CONTENTS-/5X,14A	LISTTC34
*6)	LISTTC35
GO TO 20	LISTTC36
END	LISTTC37
*ELT,I DORMAN.OPT	OPT 3
SUBROUTINE OPT	OPT 5
C	OPT 6
C	OPT 7
C LISTS THE OPTIONS AVAILABLE	OPT 8
C	OPT 9
C PROGRAMMER - S. WRAY	OPT 10
C	OPT 11
COMMON /MISC/ERFLAG,FERR,KARD(14),ACTION(14)	MISC 2
INTEGER ERFLAG,FERR,ACTION,PRTFIL	MISC 3
EQUIVALENCE (PRTFIL,FERR)	MISC 4
WRITE (PRTFIL,5)	OPT 13

5 FORMAT(25H OPTION LIST (SHORT FORM) /	OPT	14
*57H 1. CREATE 2. USE (U) 3. OPTION (O) 4. SAVE (S) /	OPT	15
*57H 5. ADD (A) 6. DELETE (D) 7. EDIT (E) 8. CONVERT (C) /	OPT	16
*57H 9. REPLACE(R) 10. LIST (L) 11. DONE 12. TAPE LIST /	OPT	17
*)	OPT	18
9 WRITE (PRTFIL,10)	OPT	19
10 FORMAT(39H ENTER NUMBER OF OPTION TO BE EXPLAINED/8H OR DONE/	OPT	20
*9H READY -)	OPT	21
15 CALL INCD (KARD,5)	OPT	22
IF(KARD(1).EQ.4HDONE) RETURN	OPT	23
CALL VALUE (KARD,V,IERR)	OPT	24
IF(IERR.EQ.0) GO TO 20	OPT	25
17 WRITE (PRTFIL,16)	OPT	26
16 FORMAT(26H ILLEGAL ENTRY - TRY AGAIN/9H READY -)	OPT	27
GO TO 15	OPT	28
20 I = V	OPT	29
IF(I.LT.1) GO TO 17	OPT	30
IF(I.GT.12) GO TO 17	OPT	31
GO TO (101,102,103,104,105,106,107,108,109,110,111,112,113	OPT	32
*),I	OPT	33
101 WRITE (PRTFIL,201)	OPT	34
201 FORMAT(71H IF THE KEY WORD -CREATE- IS ENTERED, THE PROGRAM WILL GOPT	OPT	35
*ENERATE A DUMMY/63H DATA BASE AND THEN USE THE ADD OPTION TO ADD DOPT	OPT	36
*ECKS FROM TAPE21/29H CREATE MAY BE USED ONLY ONCE)	OPT	37
GO TO 9	OPT	38
102 WRITE (PRTFIL,202)	OPT	39
202 FORMAT(71H IF THE KEY WORD -USE- IS ENTERED, THE PROGRAM WILL REQUOPT	OPT	40
*EST THE NAME OF/74H A DECK TO BE EXTRACTED FROM THE DATA BASE. THOPT	OPT	41
*ERE WILL BE A DELAY FOR THE/11H EXTRACTION)	OPT	42
GO TO 9	OPT	43
103 WRITE(PRTFIL,203)	OPT	44
203 FORMAT(70H IF THE KEY WORD -OPTION- IS ENTERED, THE USER IS PROVIDOPT	OPT	45
*ED WITH A LIST/68H AND THE DESCRIPTION OF EACH ITEM IN THE LIST ASOPT	OPT	46
* REQUESTED VIA INPUT)	OPT	47
GO TO 9	OPT	48
104 WRITE (PRTFIL,204)	OPT	49

204	FORMAT(72H IF THE KEY WORD -SAVE- IS ENTERED, THE USER WILL HAVE TOPT	50
	*HE CAPABILITY TO/71H SAVE THE MOST RECENT VERSION OF THE DATA BASEOPT	51
	* OR SAVE A BASIC DECK FOR/60H INPUT TO DORCA OR SAVE A MOD DECK TOOPT	52
	* BE RETAINED EXTERNALLY)	OPT 53
	GO TO 9	OPT 54
105	WRITE (PRTFIL,205)	OPT 55
205	FORMAT(71H IF THE KEY WORD -ADD- IS ENTERED, THE PROGRAM WILL BEGIOPT	56
	*N TO LOAD DECKS/42H FROM TAPE21 FOR ADDITION TO THE DATA BASE)	OPT 57
	GO TO 9	OPT 58
106	WRITE (PRTFIL,206)	OPT 59
206	FORMAT(71H IF THE KEY WORD -DELETE- IS ENTERED, THE PROGRAM WILL ROPT	60
	*EQUEST THE NAME/72H OF THE DECK TO BE DELETED FROM THE DATA BASE. OPT	61
	* THE DECK IS THEN DELETED)	OPT 62
	GO TO 9	OPT 63
107	WRITE (PRTFIL,207)	OPT 64
207	FORMAT(71H IF THE KEY WORD -EDIT- IS ENTERED, THE USER IS DIRECTEDOPT	65
	* TO NAME A DECK/71H IN THE DATA BASE AND THEN ENTER A MOD DECK VIAOPT	66
	* THE TERMINAL WHICH WILL/43H ALTER THE ORIGINAL DECK VIA THE USE OOPT	67
	*PTION)	OPT 68
	GO TO 9	OPT 69
108	WRITE (PRTFIL,208)	OPT 70
208	FORMAT(73H IF THE KEY WORD -CONVERT- IS ENTERED, THE PROGRAM WILL OPT	71
	*REQUEST THE NAMES/76H OF TWO DECKS IN THE DATA BASE. THE TWO DECKOPT	72
	*S WILL BE COMPARED CARD BY CARD/74H AND A MOD DECK WILL BE GENERATOPT	73
	*ED THAT WILL BE SMALLER THAN THE BASIC DECK/68H BUT EQUIVALENT IN OPT	74
	*DATA CONTENT WHEN APPLIED TO THE OTHER BASIC DECK)	OPT 75
	GO TO 9	OPT 76
109	WRITE (PRTFIL,209)	OPT 77
209	FORMAT(74H IF THE KEY WORD -REPLACE- IS ENTERED, THE USER WILL BE OPT	78
	*ALLOWED TO REPLACE/63H A DECK IN THE DATA BASE IF THERE IS NOT A MOPT	79
	*OD DECK NEEDING THE/14H ORIGINAL DECK)	OPT 80
	GO TO 9	OPT 81
110	WRITE (PRTFIL,210)	OPT 82
210	FORMAT(53H IF THE KEY WORD -LIST- IS ENTERED, THE USER CAN LIST/ OPT	83
	*36H TABLE OF CONTENTS OF THE DATA BASE /	OPT 84
	*40H GENEALOGY OF ANY DECK IN THE DATA BASE /	OPT 85

*48H COUNT OF CARDS ON ANY TAPE USED BY THE PROGRAM /	OPT	86
*50H INDIVIDUAL CARDS ON ANY TAPE USED BY THE PROGRAM /	OPT	87
*55H PRINTOUT OF THE COMPLETE CONTENTS OF ANY TAPE OR DECK)	OPT	88
GO TO 9	OPT	89
111 WRITE (PRTFIL,211)	OPT	90
211 FORMAT(58H IF THE KEY WORD - DONE- IS ENTERED, THE PROGRAM TERMINA	OPT	91
*ES)	OPT	92
GO TO 9	OPT	93
112 WRITE (PRTFIL,212)	OPT	94
212 FORMAT(44H THIS IS A LIST OF TAPESUSED BY THE PROGRAM/	OPT	95
*62H TAPE1 - SOURCE DATA BASE TAPE14 - CURRENT BASIC DECK	/OPT	96
*62H TAPE2 - ALTERNATE DATA BASE TAPE20 - OUTPUT DECK LISTINGS	/OPT	97
*62H TAPE3 - ALTERNATE DATA BASE TAPE21 - INPUT BASIC DECKS	/OPT	98
*62H TAPE4 - SAVED DATA BASE TAPE22 - SCRATCH	/OPT	99
*62H TAPE5 - CONSOLE INPUT TAPE23 - SCRATCH	/OPT	100
*62H TAPE6 - CONSOLE OUTPUT TAPE24 - SAVED MOD DECKS	/OPT	101
*62H TAPE11 - INPUT MOD DECKS TAPE25 - SCRATCH	/OPT	102
*62H TAPE12 - DORCA DATA DECK TAPE26 - SCRATCH	/OPT	103
*62H TAPE13 - CURRENT MOD DECK TAPE27 - SCRATCH	/OPT	104
*)	OPT	105
GO TO 9	OPT	106
113 WRITE (PRTFIL,213)	OPT	107
213 FORMAT(4H 13.)	OPT	108
GO TO 9	OPT	109
END	OPT	110
*ELT,I DORMAN.OUTCD	OUTCD	3
SUBROUTINE OUTCD (ICRD,IUNIT)	OUTCD	5
C PUT IN FORCE CLOSE	OUTCD	6
C	OUTCD	7
C MODE 1 - WRITE 55 CARDS	OUTCD	8
C MODE 2 - WRITE 1 CARD	OUTCD	9
C	OUTCD	10
C PROGRAMMER: VOIT	OUTCD	11
C	OUTCD	12
COMMON /REST/TABLE,USES,FILE,END,DECK,DDECK,BLANK ,BATCH	REST	2
INTEGER TABLE,USES,FILE,END,DECK,DDECK,BLANK,BATCH	REST	3

COMMON /MISC/ERFLAG,FERR,KARD(14),ACTION(14)	MISC	2
INTEGER ERFLAG,FERR,ACTION,PRTFIL	MISC	3
EQUIVALENCE (PRTFIL,FERR)	MISC	4
COMMON /WORK/IUNIT(3),IIRW(3),IUTBL(3,19),IACT(8,3),NFILES	WORK	2
COMMON /BFRS/ XXX(54)	BFRS	2
COMMON /BFRS/ IMOD1(14,55,2),IWK1(168,2),ITEMP1(84),ITEMP2(84)	BFRS	3
DIMENSION ICRD(14)	OUTCD	17
DIMENSION ITMP1(84),ITMP2(84)	OUTCD	18
EQUIVALENCE (ITMP1(1),ITEMP1(1)),(ITEMP2(1),ITMP2(1))	OUTCD	19
DATA IIX,IYY,IIZ /14,55,2/	OUTCD	20
DATA IIW /84/	OUTCD	21
IF(IUNIT.EQ.6) GO TO 8	OUTCD	22
CALL FNOBUF (IUNIT,IX,IY)	OUTCD	23
IF (IX.NE.2) GO TO 100	OUTCD	24
C	OUTCD	25
C	OUTCD	26
C	OUTCD	27
MODE EQ 2 - WRITE ONE CARD	OUTCD	28
IF (IACT(4,IY).EQ.1) GO TO 510	OUTCD	29
8 CONTINUE	OUTCD	30
DO 2 I=1,14	OUTCD	31
J=15-I	OUTCD	32
IF(ICRD(J).NE.BLANK) GO TO 3	OUTCD	33
2 CONTINUE	OUTCD	34
3 CONTINUE	OUTCD	35
IF(IUNIT.EQ.6) GO TO 10	OUTCD	36
WRITE (IUNIT,5) (ICRD(I),I=1,J)	OUTCD	37
5 FORMAT(14A6)	OUTCD	38
IF(ICRD(1).NE.END) RETURN	OUTCD	39
IF(ICRD(2).NE.FILE) RETURN	OUTCD	40
IACT(4,IY) = 1	OUTCD	41
END FILE IUNIT	OUTCD	42
RETURN	OUTCD	43
C	OUTCD	44
C	OUTCD	45
C	OUTCD	46
SPECIAL FORMAT FOR TERMINAL OUTPUT		
10 WRITE (IUNIT,11) (ICRD(I),I=1, J)		

11	FORMAT(1X,14A6)	OUTCD 47
	RETURN	OUTCD 48
C		OUTCD 49
C	MODE 1	OUTCD 50
C		OUTCD 51
100	CONTINUE	OUTCD 52
	IF(IX.NE.1) GO TO 500	OUTCD 53
	IXX = IACT(8,IY)	OUTCD 54
	IYY = IACT(7,IY)	OUTCD 55
	IF (IXX.LT.IIW) GO TO 120	OUTCD 56
C		OUTCD 57
C	MOVE 84 CHAR FROM WORK AREA INTO BUFFER	OUTCD 58
C		OUTCD 59
	CALL A1TA6 (IWK1(1,IY),IMOD1(1,IYY,IY))	OUTCD 60
	IXX = IXX-84	OUTCD 61
	IYY = IYY+1	OUTCD 62
C		OUTCD 63
C	MOVE REMAINING CHAR DOWN	OUTCD 64
C		OUTCD 65
	DO 110 I=1,IXX	OUTCD 66
	IZ = I+84	OUTCD 67
	IWK1 (1,IY) = IWK1 (IZ,IY)	OUTCD 68
110	CONTINUE	OUTCD 69
C		OUTCD 70
C	MOVE NEW CARD INTO WORK AREA	OUTCD 71
C		OUTCD 72
120	CONTINUE	OUTCD 73
	CALL A6TA1 (ICRD,ITMP1)	OUTCD 74
	CALL PACCON(ITEMP1,ITMP2,ICH)	OUTCD 75
	DO 130 I=1,ICH	OUTCD 76
	ICH1 = IXX+I	OUTCD 77
	IWK1(ICH1,IY) = ITMP2(I)	OUTCD 78
130	CONTINUE	OUTCD 79
	IXX = IXX+ICH	OUTCD 80
C		OUTCD 81
C	CHECK FOR END OF FILE	OUTCD 82

C	IAC(7,IY) = IYY	OUTCD 83
	IAC(8,IY) = IXX	OUTCD 84
	IF(ICRD(1).NE.6H\$END 0) GO TO 135	OUTCD 85
	IF(ICRD(2).EQ.6HF FILE) GO TO 150	OUTCD 86
135	CONTINUE	OUTCD 87
	IF(IYY.GT.1IY) GO TO 140	OUTCD 88
	RETURN	OUTCD 89
140	CONTINUE	OUTCD 90
	DO 145 I=1,IY	OUTCD 91
	WRITE (IUNIT,5) (IMOD1(L,I,IY),L=1,14)	OUTCD 92
145	CONTINUE	OUTCD 93
	IAC(7,IY) = 1	OUTCD 94
	RETURN	OUTCD 95
C		OUTCD 96
C	MOVE WORK AREA INTO BUFFER AND	OUTCD 97
C	BLANK REMAINING BUFFER	OUTCD 98
C		OUTCD 99
150	CONTINUE	OUTCD100
	IAC(4,IY) = 1	OUTCD101
	K = (2*IIW)-IXX	OUTCD102
	K1= 2*IIW	OUTCD103
	DO 155 I=K,K1	OUTCD104
	IWK1(I,IY) = 6H	OUTCD105
155	CONTINUE	OUTCD106
	CALL A1TA6 (IWK1(1,IY),IMOD1(1,IYY,IY))	OUTCD107
	IS = IYY+1	OUTCD108
	DO 170 I=IS,IY	OUTCD109
	DO 165 J=1,14	OUTCD110
	IMOD1(J,I,IY) = 6H	OUTCD111
165	CONTINUE	OUTCD112
170	CONTINUE	OUTCD113
	DO 180 I=1,55	OUTCD114
	WRITE (IUNIT,5) (IMOD1(L,I,IY),L=1,14)	OUTCD115
180	CONTINUE	OUTCD116
	RETURN	OUTCD117
		OUTCD118

```

C
C      ERROR EXIT
C
500  CONTINUE
      WRITE(FERR,1000) IUNIT,IX
1000 FORMAT(27H OUTCD - MODE ERROR -- UNIT,I3,5H MODE,I3)
      ERFLAG = 1
      CALL TERM
510  CONTINUE
      WRITE(FERR,1010) IUNIT
1010 FORMAT(28H OUTCD - END OF FILE ON UNIT,I3)
      ERFLAG = 1
      CALL TERM
      RETURN
      END
*ELT,I DORMAN.PACCON
      SUBROUTINE PACCON (CHAR,OCHAR,NCHAR)
C
C      PACK CARDS WITH = AND ;
C      PROGRAMMER - S. WRAY
C
      INTEGER CHAR(80),OCHAR(80),PKTAB(9,2)
      DO 5000 I=1,80
      IF(CHAR(I).EQ.1H=) CHAR(I)=1H-
      IF(CHAR(I).EQ.1H;) CHAR(I) = 1H
5000 CONTINUE
      DO 5 I = 1,80
      J = 81 - I
      IF(CHAR(J).NE.1H ) GO TO 10
5 CONTINUE
      NCHAR = 1
9      OCHAR(NCHAR)=1H;
      RETURN
10  NCHAR = 81 - I
      PKTAB(1,1) = 1
      NPK = 0

```

```

OUTCD119
OUTCD120
OUTCD121
OUTCD122
OUTCD123
OUTCD124
OUTCD125
OUTCD126
OUTCD127
OUTCD128
OUTCD129
OUTCD130
OUTCD131
OUTCD132
OUTCD133
PACCON 3
PACCON 5
PACCON 6
PACCON 7
PACCON 8
PACCON 9
PACCON10
PACCON11
PACCON12
PACCON13
PACCON14
PACCON15
PACCON16
PACCON17
PACCON18
PACCON19
PACCON20
PACCON21
PACCON22
PACCON23
PACCON24

```

```

      DO 11 I=1,8
      DO 12 J=1,10
      L = 10*I + 1 - J
      IF(L.GE.NCHAR) GO TO 14
      IF(CHAR(L).NE.1H ) GO TO 13
12  CONTINUE
      L=L-1
13  IF(J.EQ.1) GO TO 11
      CHAR(L+1) = 1H=
      NPK=NPK+1
      PKTAB(NPK,2)= L+1
      PKTAB(NPK+1,1) = 10*I +1
11  CONTINUE
      GO TO 15
14  NPK = NPK+1
      PKTAB(NPK,2) = NCHAR+1
      CHAR(NCHAR+1) = 1H;
15  NCHAR = 0
      DO 16 L=1,NPK
      L1=PKTAB(L,1)
      L2=PKTAB(L,2)
      DO 17 K=L1,L2
      NCHAR=NCHAR+1
      OCHAR(NCHAR)=CHAR(K)
17  CONTINUE
16  CONTINUE
      RETURN
      END
*ELT,I DORMAN.REPL
      SUBROUTINE REPL
C
C          CONTROLS REPLACEMENT OF DECKS
C
C          PROGRAMMER - S. WRAY
C
      COMMON /MISC/ERFLAG,FERR,KARD(14),ACTION(14)

```

```

PACCON25
PACCON26
PACCON27
PACCON28
PACCON29
PACCON30
PACCON31
PACCON32
PACCON33
PACCON34
PACCON35
PACCON36
PACCON37
PACCON38
PACCON39
PACCON40
PACCON41
PACCON42
PACCON43
PACCON44
PACCON45
PACCON46
PACCON47
PACCON48
PACCON49
PACCON50
PACCON51
PACCON52
REPL  3
REPL  5
REPL  6
REPL  7
REPL  8
REPL  9
REPL 10
MISC  2

```

INTEGER ERFLAG,FERR,ACTION,PRTFIL	MISC	3
EQUIVALENCE (PRTFIL,FERR)	MISC	4
COMMON /FILES/ BASIC,MTAPE,FINAL,S1,S2	FILES	2
INTEGER BASIC,MTAPE,FINAL,S1,S2	FILES	3
COMMON /NAMES/ IVER,DNAME(2),MODNAM(2),BNAME(2)	NAMES	2
INTEGER DNAME,MODNAM,BNAME	NAMES	3
DIMENSION IU(3),IR(3)	REPL	14
DATA IU,IR/0,0,0,2,2,2/	REPL	15
WRITE(PRTFIL,5)	REPL	16
5 FORMAT(30H ENTER NAME OF DECK TO BE USED/9H READY -)	REPL	17
CALL INCD (KARD,5)	REPL	18
M1 = 13	REPL	19
I = 0	REPL	20
M2 = 14	REPL	21
IF((KARD(1).EQ.MODNAM(1)).AND.(KARD(2).EQ.MODNAM(2))) I = M1	REPL	22
IF((KARD(1).EQ.BNAME(1)).AND.(KARD(2).EQ.BNAME(2))) I = M2	REPL	23
IF(I.NE.0) GO TO 20	REPL	24
WRITE (PRTFIL,15)	REPL	25
15 FORMAT(24H ORIGINAL DECK NOT FOUND)	REPL	26
ERFLAG = 1	REPL	27
RETURN	REPL	28
20 CONTINUE	REPL	29
WRITE (PRTFIL,10)	REPL	30
10 FORMAT(34H ENTER NAME OF DECK TO BE REPLACED/9H READY -)	REPL	31
CALL INCD (ACTION,5)	REPL	32
IF((KARD(1).EQ.ACTION(1)).AND.(KARD(2).EQ.ACTION(2))) GO TO 100	REPL	33
WRITE (PRTFIL,25)	REPL	34
25 FORMAT(24H DECK NAMES DO NOT MATCH/38H REQUEST REJECTED - USE DELETE AND ADD)	REPL	35
ERFLAG = 1	REPL	36
RETURN	REPL	37
100 CONTINUE	REPL	38
S1 = BASIC + 1	REPL	39
IF(S1.GT.3) S1 = 2	REPL	40
IU(1) = BASIC	REPL	41
IU(2) = S1	REPL	42
	REPL	43

```

        IU(3) = I
        REWIND I
        REWIND S1
        CALL INTBUF(IU,IR)
        CALL REPLAC(BASIC,I,S1)
        IF(ERFLAG.NE.C) RETURN
        BASIC = S1
        RETURN
    END
#ELT,I DORMAN.REPLAC
    SUBROUTINE REPLAC(IN1,IN2,OUT)
C SUBSTITUTE DECK WHICH IS NEXT ON FILE IN2 FOR THE DECK OF THE SAME
C NAME WHICH IS ON THE BASIC DATA FILE IN1. COPY THE REVISED BASIC DATA
C FILE ONTO FILE OUT.
C PROGRAMMER - B. GOLD
C
    COMMON /MISC/ERFLAG,FERR,KARD(14),ACTION(14)
    INTEGER ERFLAG,FERR,ACTION,PRTFIL
    EQUIVALENCE (PRTFIL,FERR)
    INTEGER OUT
C
    INTEGER TYPE, FLAG, BASIK, MOD, TEMP(14)
    COMMON /REST/TABLE,USES,FILE,END,DECK,DDECK,BLANK ,BATCH
    INTEGER TABLE,USES,FILE,END,DECK,DDECK,BLANK,BATCH
    INTEGER VERS
    DATA VERS/6HVERSIO/
    DATA BASIK, MOD / 5HBASIC, 3HMOD /
C
C INITIALIZE FILES. READ FIRST CARD OF NEW DECK AND VERSION CARD FROM
C DATA FILE.
C
    FLAG = 1
    REWIND IN1
    REWIND OUT
    ERFLAG = 0
    CALL INCD(KARD,IN2)

```

```

REPL 44
REPL 45
REPL 46
REPL 47
REPL 48
REPL 49
REPL 50
REPL 51
REPL 52
REPLAC 3
REPLAC 5
REPLAC 6
REPLAC 7
REPLAC 8
REPLAC 9
REPLAC10
MISC 2
MISC 3
MISC 4
REPLAC12
REPLAC13
REPLAC14
REST 2
REST 3
REPLAC16
REPLAC17
REPLAC18
REPLAC19
REPLAC20
REPLAC21
REPLAC22
REPLAC23
REPLAC24
REPLAC25
REPLAC26
REPLAC27

```

KOUNT2 = 1		REPLAC28
IF (ERFLAG.NE.0) GO TO 100		REPLAC29
TYPE = BASIK		REPLAC30
IF (KARD(4).EQ.USES) TYPE = MOD		REPLAC31
IF (KARD(1).NE.DDECK) GO TO 110		REPLAC32
NAME1 = KARD(2)		REPLAC33
NAME2 = KARD(3)		REPLAC34
CALL EXTRAN (KARD(7),KARD(9))		REPLAC35
WRITE (PRTFIL,10) KARD(2), KARD(3)		REPLAC36
10 FORMAT(20HOREPLACE DECK NAMED ,2A6,13H ON DATA FILE/)		REPLAC37
CALL OUTCU (KARD,PRTFIL)		REPLAC38
IF ((KARD(2).EQ.KARD(5)).AND.(KARD(3).EQ.KARD(6))) GO TO 170		REPLAC39
KOUNT1= 0		REPLAC40
C		REPLAC41
C COPY TABLE OF CONTENTS. CHECK TO SEE THAT REPLACEMENT DECK AND ANY		REPLAC42
C DECK IT USES ARE LISTED AMONG CONTENTS.		REPLAC43
C		REPLAC44
FLAG = 2		REPLAC45
INDEX = 0		REPLAC46
NEED = 0		REPLAC47
LASTM = 0		REPLAC48
NTC = 0		REPLAC49
IUSE = 0		REPLAC50
30 CALL INCD(TEMP,1N1)		REPLAC51
IF (ERFLAG.NE.0) GO TO 120		REPLAC52
KOUNT1 = KOUNT1+1		REPLAC53
IF (TEMP(1).EQ.END) GO TO 40		REPLAC54
IF(TEMP(5).EQ.VERS) GO TO 30		REPLAC55
IF (TEMP(1).NE.DDECK) GO TO 150		REPLAC56
NTC = NTC+1		REPLAC57
IF (TEMP(4).EQ.USES) LASTM = NTC		REPLAC58
IF ((KARD(2).EQ.TEMP(5)).AND.(KARD(3).EQ.TEMP(6))) IUSE = NTC		REPLAC59
IF ((TEMP(2).EQ.KARD(5)).AND.(TEMP(3).EQ.KARD(6))) NEED = NTC		REPLAC60
IF ((TEMP(2).EQ.KARD(2)).AND.(TEMP(3).EQ.KARD(3))) INDEX= NTC		REPLAC61
GO TO 30		REPLAC62
C		REPLAC63

C DECIDE WHERE TO INSERT NEW \$DECK CARD IN TABLE OF CONTENTS

C

40 LOC = INDEX

IF (INDEX.EQ.0) GO TO 130

IF ((NEED.EQ.0).AND.(TYPE.EQ.MOD)) GO TO 140

IF ((NEED.GT.0).AND.(NEED.LT.IUSE)) GO TO 160

INDEX = IUSE+1

IF (TYPE.EQ.BASIK) INDEX = NTC

C

C READ TABLE OF CONTENTS AGAIN AND COPY ONTO OUT FILE WITH NEW \$DECK

C CARD INSERT AND OLD ONE DELETED

C

REWIND IN1

KOUNT1 = 0

KOUNT3 = 0

45 CALL INCD(TEMP,IN1)

IF (ERFLAG.NE.0) GO TO 120

KOUNT1 = KOUNT1+1

IF ((TEMP(2).EQ.NAME1).AND.(TEMP(3).EQ.NAME2)) GO TO 45

IF (INDEX.NE.KOUNT3) GO TO 47

CALL OUTCD(KARD,OUT)

KOUNT3 = KOUNT3+1

47 CALL OUTCD(TEMP,OUT)

KOUNT3 = KOUNT3+1

IF ((TEMP(1).NE.END).OR.(TEMP(2).NE.TABLE)) GO TO 45

C

C COPY ALL DECKS FROM IN1 TO OUT EXCEPT DECK TO BE REPLACED.

C INSERT NEW DECK WHEN DECK COUNT NDECK = NUMBER

C

FLAG = 3

NDECK = 0

IF ((TYPE.EQ.MOD) .AND. (LOC.LE.LASTM))

NUMBER = LASTM-INDEX+1

IF ((TYPE.EQ.MOD) .AND. (LOC.GT.LASTM))

NUMBER = LASTM-INDEX+2

IF ((TYPE.EQ.BASIK).AND.(LOC.LE.LASTM))

NUMBER = LASTM

IF ((TYPE.EQ.BASIK).AND.(LOC.GT.LASTM))

NUMBER = LASTM+1

50 CALL INCD(TEMP,IN1)

REPLAC64

REPLAC65

REPLAC66

REPLAC67

REPLAC68

REPLAC69

REPLAC70

REPLAC71

REPLAC72

REPLAC73

REPLAC74

REPLAC75

REPLAC76

REPLAC77

REPLAC78

REPLAC79

REPLAC80

REPLAC81

REPLAC82

REPLAC83

REPLAC84

REPLAC85

REPLAC86

REPLAC87

REPLAC88

REPLAC89

REPLAC90

REPLAC91

REPLAC92

REPLAC93

REPLAC94

REPLAC95

REPLAC96

REPLAC97

REPLAC98

REPLAC99

IF (ERFLAG.NE.0) GO TO 120	REPLA100
KOUNT1 = KOUNT1 + 1	REPLA101
IF (TEMP(1).NE.DDECK) GO TO 80	REPLA102
ISKIP = 0	REPLA103
IF ((TEMP(2).EQ.NAME1).AND.(TEMP(3).EQ.NAME2)) ISKIP = 1	REPLA104
IF (ISKIP.EQ.1) GO TO 80	REPLA105
NDECK = NDECK+1	REPLA106
IF (NDECK.NE.NUMBER) GO TO 80	REPLA107
C	REPLA108
C TRANSFER NEW DECK ONTO OUT FILE	REPLA109
C	REPLA110
55 FLAG = 4	REPLA111
CALL OUTCD(KARD,OUT)	REPLA112
60 CALL INCD(KARD,IN2)	REPLA113
IF (ERFLAG.NE.0) GO TO 100	REPLA114
KOUNT2 = KOUNT2+1	REPLA115
CALL OUTCD(KARD,OUT)	REPLA116
IF (KARD(1).NE.END) GO TO 60	REPLA117
WRITE (PRTFIL,70) KOUNT2	REPLA118
70 FORMAT(/I10,6H CARDS)	REPLA119
FLAG = 5	REPLA120
C	REPLA121
C DO NOT COPY THIS CARD ONTO OUT FILE IF IT IS PART OF THE OLD DECK TO	REPLA122
C BE REPLACED (IF ISKIP=1)	REPLA123
C	REPLA124
80 IF (ISKIP.EQ.0) CALL OUTCD(TEMP,OUT)	REPLA125
IF ((TEMP(1).NE.END).OR.(TEMP(2).NE.FILE)) GO TO 50	REPLA126
IF (ISKIP.EQ.1) CALL OUTCD(TEMP,OUT)	REPLA127
RETURN	REPLA128
C	REPLA129
C ERROR STOPS	REPLA130
C	REPLA131
100 WRITE (FERR,105)	REPLA132
105 FORMAT(23H0ERROR READING FILE IN2)	REPLA133
GO TO 200	REPLA134
110 WRITE (FERR,115)	REPLA135

115	FORMAT(37H0FIRST CARD ON FILE IN2 IS NOT \$DECK)	REPLA136
	GO TO 200	REPLA137
120	WRITE (FERR,125)	REPLA138
125	FORMAT(23H0ERROR READING FILE IN1)	REPLA139
	GO TO 200	REPLA140
130	WRITE (FERR,135) KARD(2), KARD(3)	REPLA141
135	FORMAT(22H0DECK TO BE REPLACED (,2A6,36H) IS NOT LISTED IN TABLE OF	REPLA142
	*F CONTENTS)	REPLA143
	GO TO 200	REPLA144
140	WRITE (FERR,145) (KARD(I), I=1,6)	REPLA145
145	FORMAT(52H0DECK TO BE ADDED USES DECK NOT IN TABLE OF CONTENTS/	REPLA146
	* 5X,6A6)	REPLA147
	GO TO 200	REPLA148
150	WRITE (FERR,155)	REPLA149
155	FORMAT(50H0ILLEGAL CARD WITHIN TABLE OF CONTENTS ON FILE IN1)	REPLA150
	GO TO 200	REPLA151
160	WRITE (FERR,165) (NAME1,NAME2, J=1,3)	REPLA152
165	FORMAT(63H0ERROR IN DECK POSITIONS DISCOVERED DURING REPLACEMENT OF	REPLA153
	1F DECK 2A6/17H DECK WHICH USES 2A6,22H FOLLOWS DECK USED BY 2A6)	REPLA154
	GO TO 200	REPLA155
170	WRITE (FERR,175)	REPLA156
175	FORMAT(17H0DECK USES ITSELF)	REPLA157
C		REPLA158
	200 ERFLAG = 1	REPLA159
C		REPLA160
C	ADVANCE FILE IN2 TO END OF CURRENT DECK	REPLA161
C		REPLA162
	ERFLAG = 0	REPLA163
210	IF (KARD(1).EQ.END) GO TO 220	REPLA164
	CALL INCD(KARD,IN2)	REPLA165
	KOUNT2 = KOUNT2 + 1	REPLA166
	IF (ERFLAG.EQ.0) GO TO 210	REPLA167
	WRITE (FERR,105)	REPLA168
220	ERFLAG = 1	REPLA169
	RETURN	REPLA170
	END	REPLA171

#ELT,I DORMAN.RESTOR	RESTORM3
SUBROUTINE RESTOR (BUF,NC,NL,FILE,NCNOW)	RESTORM5
C	RESTORM6
C	RESTORM7
C	RESTORM8
C	RESTORM9
C	RESTOR10
C	RESTOR11
C	RESTOR12
C	RESTOR13
C	RESTOR14
C	RESTOR15
C	RESTOR16
C	RESTOR17
C	RESTOR18
C	RESTOR19
C	RESTOR20
UPON ENTRY	WORK 2
BUF	BFRS1 2
NC	BFRS1 3
NL	BFRS1 4
FILE	BFRS1 5
NCNOW	BFRS1 6
INTEGER MAX	RESTOR22
INTEGER END	RESTOR23
INTEGER BUF,FILE	RESTOR24
DIMENSION BUF(14,20)	RESTOR25
COMMON /WORK/IIUNT(3),IIRW(3),IUTBL(3,19),IACT(8,3),NFILES	RESTOR26
COMMON /BFRS/ ISYN(2,20),ISYN1,IT1,IT2,IT3,FULL,CN1,CN2,LIN1,LIN2	RESTOR27
*,NWBUF,NB1,NB2,ICN1,ICN2	RESTOR28
COMMON /BFRS/ BUF1(14,50),BUF2(14,50),AAA(700)	RESTOR29
INTEGER CN1,CN2,BUF1,BUF2	RESTOR30
LOGICAL FULL	RESTOR31
EQUIVALENCE (NWBUF,MAX)	GWT3 2
IF (NC.EQ.NCNOW) RETURN	GWT3 3
NNG = NC	RESTOR32
NNOW = NCNOW	
END = NL	
IF (NNG.GT.NNOW) GO TO 100	
C	
C	
C	
CARD HAS PREVIOUSLY BEEN READ	
REWIND FILE	
CALL FNDBUF(FILE,IX,IY)	
IACT(4,IY) = 0	
NNG = 1	

```

      NNOW= 1
10    CONTINUE
      CALL FILBUF(1,END,MAX,BUF,FILE)
      NL = END
      IF (NC.EQ.NNC) RETURN
100   CONTINUE
      IF (NC.LE.NNOW+END-1) GO TO 120
      NNC = NNOW+END
      NNOW= NNC
      GO TO 10
120   CONTINUE
      N = NC-NNOW
C     MOVE CARDS UP, PLACING NC INTO POSITION 1
      K = END - N
      DO 140 I=1,K
      L= N+I
      DO 130 J=1,14
      BUF(J,I) = BUF(J,L)
130   CONTINUE
140   CONTINUE
      NL = K
      IF (BUF(1,K).EQ.6H$END 0) RETURN
      K = K+1
      CALL FILBUF(K,END,MAX,BUF,FILE)
      NL = END
      RETURN
      END
*ELT,I DORMAN.SAVER
      SUBROUTINE SAVER
C
C     SAVE NECESSARY FILES/DECKS
C
C     PROGRAMMER - S. WRAY
C
      DIMENSION IU(3),IR(3)
      COMMON /MISC/ERFLAG,FERR,KARD(14),ACTION(14)

```

```

RESTOR33
GWT13 5
GWT13 6
GWT13 7
RESTOR36
RESTOR37
RESTOR38
RESTOR39
RESTOR40
RESTOR41
RESTOR42
RESTOR43
RESTOR44
GWT13 8
RESTOR46
RESTOR47
RESTOR48
RESTOR49
RESTOR50
RESTOR51
RESTOR52
RESTOR53
RESTOR54
GWT13 9
GWT13 10
RESTOR58
RESTOR59
SAVER 3
SAVER 5
SAVER 6
SAVER 7
SAVER 8
SAVER 9
SAVER 10
SAVER 11
MISC 2

```

```

INTEGER ERFLAG,FERR,ACTION,PRTFIL
EQUIVALENCE (PRTFIL,FERR)
COMMON /FILES/ BASIC,MTAPE,FINAL,S1,S2
INTEGER BASIC,MTAPE,FINAL,S1,S2
COMMON /NAMES/ IVER,DNAME(2),MODNAM(2),BNAME(2)
INTEGER DNAME,MODNAM,BNAME
COMMON /REST/TABLE,USES,FILE,END,DECK,DDECK,BLANK ,BATCH
INTEGER TABLE,USES,FILE,END,DECK,DDECK,BLANK,BATCH
IU(1) = 5
IR(1) = 2
IR(2) = 2
IR(3) = 2
WRITE (PRTFIL,5)
5 FORMAT(49H ENTER SAVE OPTION (DATA BASE, DORCA OR MOD DECK)
*/9H READY - )
CALL INCD(KARD,5)
IF(KARD(1).EQ.6HDATA B) GO TO 10
IF(KARD(1).EQ.6HDORCA ) GO TO 20
IF(KARD(1).EQ.6HMOD DE ) GO TO 30
WRITE (PRTFIL,16)
16 FORMAT(40H COMMAND NOT UNDERSTOOD - PLEASE RETRY )
RETURN
10 CONTINUE
IF(BASIC.NE.1) GO TO 6
WRITE (PRTFIL,7)
7 FORMAT(31H FILE DOES NOT NEED TO BE SAVED)
RETURN
6 CONTINUE
WRITE (PRTFIL,8)
8 FORMAT(25H ENTER VERSION IDENTIFIER/9H READY - )
CALL INCD (ACTION,5)
IU(1) = 5
IU(2) = BASIC
IU(3) = 4
CALL INT3UF(IU,IR)
IVER = ACTION(1)

```

```

MISC 3
MISC 4
FILES 2
FILES 3
NAMES 2
NAMES 3
REST 2
REST 3
SAVER 16
SAVER 17
SAVER 18
SAVER 19
SAVER 20
SAVER 21
SAVER 22
SAVER 23
SAVER 24
SAVER 25
SAVER 26
SAVER 27
SAVER 28
SAVER 29
SAVER 30
SAVER 31
SAVER 32
SAVER 33
SAVER 34
SAVER 35
SAVER 36
SAVER 37
SAVER 38
SAVER 39
SAVER 40
SAVER 41
SAVER 42
SAVER 43

```

CALL LABLER(IVER,BASIC,4)	SAVER 44
RETURN	SAVER 45
20 CONTINUE	SAVER 46
FINAL = 14	SAVER 47
WRITE (PRTFIL,21)	SAVER 48
21 FORMAT(46H ENTER NAME OF DECK TO BE USED FOR LORCA INPUT/9H READY *-)	SAVER 49
CALL INCD (KARD,5)	SAVER 50
IF((KARD(1).EQ.BNAME(1)).AND.(KARD(2).EQ.BNAME(2))) GO TO 40	SAVER 51
IF(BNAME(1).EQ.0) GO TO 38	SAVER 52
37 WRITE (PRTFIL,39)	SAVER 53
39 FORMAT(18H DECK IS NOT BASIC/29H DESTRUCT PERMISSION REQUIRED/	SAVER 54
*16H ENTER YES OR NO/9H READY -)	SAVER 55
CALL INCD(ACTION,5)	SAVER 56
IF(ACTION(1).EQ.2HNO) RETURN	SAVER 57
IF(ACTION(1).NE.3HYES) GO TO 37	SAVER 58
38 CONTINUE	SAVER 59
ACTION(1) = KARD(1)	SAVER 60
ACTION(2) = KARD(2)	SAVER 61
CALL USE (ACTION)	SAVER 62
40 CONTINUE	SAVER 63
REWIND FINAL	SAVER 64
IU(1) = 5	SAVER 65
IU(2) = FINAL	SAVER 66
IU(3) = 12	SAVER 67
CALL INTBUF(IU,IR)	SAVER 68
REWIND 12	SAVER 69
REWIND FINAL	SAVER 70
22 CALL INCD (KARD,FINAL)	SAVER 71
IF(ERFLAG.NE.0) RETURN	SAVER 72
IF(KARD(1).EQ.DDECK) GO TO 22	SAVER 73
IF(KARD(1).EQ.END) GO TO 23	SAVER 74
CALL OUTCD(KARD,12)	SAVER 75
GO TO 22	SAVER 76
23 CONTINUE	SAVER 77
DNAME (1) = ACTION(1)	SAVER 78
	SAVER 79

DNAME (2) = ACTION(2)	SAVER 80
REWIND 12	SAVER 81
RETURN	SAVER 82
30 CONTINUE	SAVER 83
WRITE (PRTFIL,51)	SAVER 84
51 FORMAT(43H ENTER NAME OF DECK TO BE SAVED AS MOD DECK/9H READY -)	SAVER 85
CALL INCD (ACTION,5)	SAVER 86
IF((ACTION(1).EQ.MODNAM(1)).AND.(ACTION(2).EQ.MODNAM(2))) GO TO 60	SAVER 87
WRITE (PRTFIL,52)	SAVER 88
52 FORMAT(24H DECK NAMES DO NOT MATCH/23H MOD DECK NOT AVAILABLE/	SAVER 89
*15H REQUEST DENIED)	SAVER 90
ERFLAG = 1	SAVER 91
RETURN	SAVER 92
60 CONTINUE	SAVER 93
IU(1) = 13	SAVER 94
IU(2) = 24	SAVER 95
CALL INTBUF(IU,IR)	SAVER 96
M1 = 24	SAVER 97
REWIND M1	SAVER 98
M2 = 13	SAVER 99
REWIND M2	SAVER100
65 CALL INCD (KARD,M2)	SAVER101
IF(ERFLAG.NE.0) RETURN	SAVER102
CALL OUTCD (KARD,M1)	SAVER103
IF(KARD(1).NE.END) GO TO 65	SAVER104
RETURN	SAVER105
END	SAVER106
*ELT,I DORMAN.SYNCBF	SYNCBFM3
SUBROUTINE SYNCBF (BUF1,NC1,NL1,FILE1,NSYNC1,	SYNCBFM5
* ISYN1,	SYNCBFM6
* BUF2,NC2,NL2,FILE2,NSYNC2)	SYNCBFM7
COMMON /BUFFER/ MAX	SYNCBFM8
INTEGER BUF1,BUF2,FILE1,FILE2	SYNCBFM9
COMMON /BFRS/	SYNCBF10
* ISYN(2,20)	SYNCBF11
DIMENSION BUF1(14,20),BUF2(14,20)	SYNCBF12

```

C      INTEGER END
C
C      NCC11 - FIRST CARD IN BUFFER 1
C      NCC21 - FIRST CARD IN BUFFER 2
C
C      NCC1 - NCC2 CURRENT CARD BEING CONSIDERED
C      NLL1 - NLL2 NR OF CARDS IN BUFFER
C      NCB1 - NCB2 PRESENT INDEX IN BUFFER
C      ISYN1 - INDEX TO THE SYNC CARDS
C
      NCC11= NC1
      NCC21= NC2
      NFLG1 = 0
      NFLG2 = 0
      NCC1 = 1
      NCC2 = 1
      NLL1 = NL1
      NLL2 = NL2
      NCB1 = 1
      NCB2 = 1
      NSYN1 = ISYN(1,ISYN1) -NC1 + 1
      NSYN2 = ISYN(2,ISYN1) -NC2 + 1
      GO TO 70
10     CONTINUE
      IF(BUF2(1,NLL2).NE.6H$END 0) GO TO 30
15     NSYNC1 = NCC1
      NSYNC2 = NCC2
      CALL RESTOR(BUF1,NC1,NLL1,FILE1,NCC11)
      CALL RESTOR(BUF2,NC2,NLL2,FILE2,NCC21)
      RETURN
30     IF(IFLG2.EQ.1) GO TO 15
      CALL FILBUF(1,END,MAX,BUF2,FILE2)
      NCC21 = NCC21 + NLL2
      NCC2 = NCC21 - NC2 + 1
      NLL2 = END
      NCB2 = 1

```

```

SYNCBF13
SYNCBF14
SYNCBF15
SYNCBF16
SYNCBF17
SYNCBF18
SYNCBF19
SYNCBF20
SYNCBF21
SYNCBF22
SYNCBF23
SYNCBF24
GWT13 11
GWT13 12
GWT13 13
SYNCBF26
SYNCBF27
SYNCBF28
SYNCBF29
SYNCBF30
SYNCBF31
SYNCBF32
GWT13 14
GWT13 15
GWT13 16
GWT13 17
GWT13 18
GWT13 19
GWT13 20
GWT13 21
GWT13 22
GWT13 23
GWT13 24
GWT13 25
GWT13 26
GWT13 27

```


	CALL RESTOR(BUF1,NC1,NLL1,FILE1,NCC11)	GWT13 28
	NCC11 = NC1	GWT13 29
	NCB1 = 1	GWT13 30
	NCC1 = 1	GWT13 31
	NLL1 = NLL1	GWT13 32
	IFLG1 = 0	GWT13 33
	GO TO 40	GWT13 34
40	CONTINUE	GWT13 35
	DO 45 I = 1,14	GWT13 36
	IF(BUF1(I,NCB1).NE.BUF2(I,NCB2)) GO TO 50	GWT13 37
45	CONTINUE	GWT13 38
	GO TO 15	GWT13 39
50	NCB2 = NCB2 + 1	GWT13 40
	NCC2 = NCC2 + 1	GWT13 41
	IF(NSYN2.NE.NCC2) GO TO 55	GWT13 42
	IFLG2 = 1	GWT13 43
	GO TO 60	GWT13 44
55	IF(NCB2.LT.NLL2) GO TO 40	GWT13 45
60	CONTINUE	GWT13 46
	NCB1 = NCB1 + 1	GWT13 47
	NCC1 = NCC1 + 1	GWT13 48
	IF(NSYN1.NE.NCC1) GO TO 65	GWT13 49
	IFLG1 = 1	GWT13 50
	GO TO 10	GWT13 51
65	IF(NCB1.LT.NLL1) GO TO 85	GWT13 52
70	IF(BUF1(1,NLL1).NE.6H\$END 0) GO TO 80	GWT13 53
	GO TO 10	GWT13 54
80	CONTINUE	GWT13 55
	CALL FILBUF(1,END,MAX,BUF1,FILE1)	GWT13 56
	NCB1 = 1	GWT13 57
	NCC11 = NCC11 + NLL1	GWT13 58
	NCC1 = NCC11 - NC1 + 1	GWT13 59
	NLL1 = END	GWT13 60
85	CALL RESTOR(BUF2,NCC21,NLL2,FILE2,NCC21)	GWT13 61
	NCB2 = 1	GWT13 62
	NCC2 = NCC21 - NC2 + 1	GWT13 63

```

        IFLG2 = 0
        GO TO 40
        END
*ELT,I  DORMAN.SYNCD5
        SUBROUTINE SYNCD5 (FSYN)
        COMMON /BFRS/ ISYN(2,20)
        INTEGER FSYN,KARD(16),ITMP(16)

C
C      READ IN SYNC CARDS
C
        DIMENSION IUNT(3),IRW(3)
        IUNT(1)=FSYN
        IUNT(2)= 0
        IUNT(3)= 0
        IRW(1) = 0
        IRW(2) = 0
        IRW(3) = 1
        DO 1 I=1,20
        DO 4 J=1,2
        ISYN(J,I) = 100000
+      CONTINUE
1      CONTINUE
        CALL INTBUF(IUNT,IRW)
        WRITE (6,100)
100  FORMAT(17H INPUT SYNC CARDS)
        DO 2 I=1,20
105  CONTINUE
        WRITE (6,110)
110  FORMAT(9H READY - )
        CALL INCD (KARD,FSYN)
        IF(KARD(1).EQ.4HDONE) GO TO 3
        ENCODE (84,62,ITMP) (KARD(J),J=1,14)
        62  FORMAT(14A6)
        DECODE(80,64,ITMP) KARD
        64  FORMAT(8(A6,A4))
        CALL VALUE (KARD,A,IERR)

```

```

GWT13 64
GWT13 65
SYNCB135
SYNCDS 3
SYNCDS 5
SYNCDS 6
SYNCDS 7
SYNCDS 8
SYNCDS 9
SYNCDS10
SYNCDS11
SYNCDS12
SYNCDS13
SYNCDS14
SYNCDS15
SYNCDS16
SYNCDS17
SYNCDS18
SYNCDS19
SYNCDS20
SYNCDS21
SYNCDS22
SYNCDS23
SYNCDS24
SYNCDS25
SYNCDS26
SYNCDS27
SYNCDS28
SYNCDS29
SYNCDS30
SYNCDS31
SYNCDS32
SYNCDS33
SYNCDS34
SYNCDS35
SYNCDS36

```

IF(IERR.GT.0) GO TO 520	SYNCD537
ISYN(1,I) = A	SYNCD538
CALL VALUE (KARD(3),A,IERR)	SYNCD539
IF(IERR.GT.0) GO TO 520	SYNCD540
ISYN(2,I) = A	SYNCD541
GO TO 17	SYNCD542
520 CONTINUE	SYNCD543
WRITE (6,530)	SYNCD544
530 FORMAT(37H BAD NUMERIC ENTRY - PLEASE REED CARD)	SYNCD545
GO TO 105	SYNCD546
17 CONTINUE	SYNCD547
2 CONTINUE	SYNCD548
3 CONTINUE	SYNCD549
RETURN	SYNCD550
END	SYNCD551
*ELT,I DORMAN.SYNC1	SYNC1 3
SUBROUTINE SYNC1	SYNC1 5
COMMON /BFRS/ ISYN(2,20),ISYN1,IT1,IT2,IT3,FULL,CN1,CN2,LIN1,LIN2	BFRS1 2
* ,NWBUF,NB1,NB2,ICN1,ICN2	BFRS1 3
COMMON /BFRS/ BUF1(14,50),BUF2(14,50),AAA(700)	BFRS1 4
INTEGER CN1,CN2,BUF1,BUF2	BFRS1 5
LOGICAL FULL	BFRS1 6
INTEGER FIN1,FIN2	SYNC1 7
INTEGER FOUT	SYNC1 8
EQUIVALENCE (FIN1,IT1)	SYNC1 9
EQUIVALENCE (FIN2,IT2)	SYNC1 10
EQUIVALENCE (FOUT,IT3)	SYNC1 11
INTEGER PRTFIL	SYNC1 12
DATA PRTFIL/6/	SYNC1 13
C	SYNC1 15
C FILES NOW OUT OF SYNC, RECOVER	SYNC1 16
C	SYNC1 17
21 CONTINUE	SYNC1 18
FULL = .TRUE.	SYNC1 19
IF(CN1.EQ.1) GO TO 30	SYNC1 21
J=1	SYNC1 22

```

        DO 25 L=CN1,NB1
        DO 22 I=1,14
22      BUF1(I,J)=BUF1(I,L)
25      J=J+1
        NB1 = J - 1
        CN1 = 1
        IF(BUF1(1,NB1).EQ.6H$END 0) GO TO 30
28      CONTINUE
        CALL FILBUF(J,NB1,NWBUF,BUF1,FIN1)
30      CONTINUE
        IF(CN2.EQ.1) GO TO 40
        J=1
        DO 35 L=CN2,NB2
        DO 32 I=1,14
32      BUF2(I,J)=BUF2(I,L)
35      J=J+1
        NB2 = J - 1
        CN2 = 1
        IF(BUF2(1,NB2).EQ.6H$END 0) GO TO 40
38      CONTINUE
        CALL FILBUF(J,NB2,NWBUF,BUF2,FIN2)
40      CONTINUE
C
C   ATTEMPT TO MATCH IN BUFFER
C
10      CONTINUE
        N1 = ICN1 - LIN1 + 1
        N2 = ICN2 - LIN2 + 1
        IS=0
        NBX = NB1
        IF(NBX.GT.N1) NBX = N1
        NBY = NB2
        IF(NBY.GT.N2) NBY = N2
        DO 60 K=CN1,NBX
        LL=K-CN1+CN2
        IF(LL.GT.NBY) LL = NBY

```

```

SYNC1 23
SYNC1 24
SYNC1 25
SYNC1 26
GWT13 66
GWT13 67
GWT13 68
GWT13 69
SYNC1 27
SYNC1 29
SYNC1 31
SYNC1 32
SYNC1 33
SYNC1 34
SYNC1 35
SYNC1 36
GWT13 70
GWT13 71
GWT13 72
GWT13 73
SYNC1 37
SYNC1 39
SYNC1 40
SYNC1 41
SYNC1 42
SYNC1 43
SYNC1 44
SYNC1 45
SYNC1 46
SYNC1 47
SYNC1 48
SYNC1 49
SYNC1 50
SYNC1 51
SYNC1 52
SYNC1 53

```

```

DO 55 L=CN2,LL
IF(BUF1(1,K).NE.BUF2(1,L)) GO TO 55
DO 50 I=2,14
IF(BUF1(I,K).NE.BUF2(I,L)) GO TO 55
50 CONTINUE
GO TO 70
55 CONTINUE
60 CONTINUE
K = NBX
L = NBY
IS= 1
70 CONTINUE
DO 73 N=CN2,NBY
NN=N-CN2+CN1
IF(NN.GT.NBX) NN = NBX
DO 72 M=CN1,NN
IF(BUF1(1,M).NE.BUF2(1,N)) GO TO 72
DO 71 I = 2,14
IF(BUF1(I,M).NE.BUF2(I,N)) GO TO 72
71 CONTINUE
GO TO 171
72 CONTINUE
73 CONTINUE
M = NBX
N = NBY
IF(IS.EQ.0) GO TO 74
GO TO 300
171 CONTINUE
IF((K+L).LE.(M+N)) GO TO 74
L=N
K=M
74 CONTINUE
C
C      USE SYN CARUS
C
IF((K.LT.N1).AND.(L.LT.N2)) GO TO 100

```

```

SYNO1 54
SYNO1 55
SYNO1 56
SYNO1 57
SYNO1 58
SYNO1 59
SYNO1 60
SYNO1 61
GWT13 74
GWT13 75
GWT13 76
SYNO1 64
SYNO1 65
SYNO1 66
SYNO1 67
SYNO1 68
SYNO1 69
SYNO1 70
SYNO1 71
SYNO1 72
GWT 1
SYNO1 78
SYNO1 79
GWT13 77
GWT13 76
SYNO1 81
SYNO1 82
GWT 2
GWT 3
GWT 4
GWT 5
SYNO1 83
SYNO1 84
SYNO1 85
SYNO1 86
SYNO1 87

```

CALL CKSYN(ICN1,ICN2,ISYN1,1)	SYN01 88
100 CONTINUE	SYN01 89
IF(K.NE.CN1) GO TO 80	SYN01 90
C	SYN01 91
C MUST DO INSERT	SYN01 92
C	SYN01 93
CALL INSERT (L+LIN2-CN2,0)	SYN01 94
RETURN	SYN01 95
C	SYN01 96
C MUST DO DELETE	SYN01 97
C	SYN01 98
30 CONTINUE	SYN01 99
CALL DELCD (K-CN1+LIN1)	SYN1100
IF(L.NE.CN2) CALL INSERT (LIN2+L-CN2,1)	SYN1101
RETURN	SYN1102
300 CONTINUE	SYN1103
WRITE (PRTFIL,9)	SYN1104
9 FORMAT(54H THERE WILL BE A WAIT FOR OUT OF CORE SEARCH FOR MATCH)	SYN1105
CALL SYNCBF(BUF1,LIN1,NB1,FIN1,K,ISYN1,	GWT+ 1
* BUF2,LIN2,NB2,FIN2,L)	GWT+ 2
CALL SYNCBF(BUF2,LIN2,NB2,FIN2,N,ISYN1,	GWT+ 3
* BUF1,LIN1,NB1,FIN1,M)	GWT+ 4
GO TO 171	SYN1110
END	SYN1111
*ELT,I DORMAN.UNPAC	UNPAC 3
SUBROUTINE UNPAC(A,B,N)	UNPAC 5
C	UNPAC 6
C REMOVE = SIGNS AND REPLACE WITH BLANKS	UNPAC 7
C REMOVE ;	UNPAC 8
C	UNPAC 9
C PROGRAMMER: VUIT	UNPAC 10
C	UNPAC 11
COMMON /MISC/ERFLAG,FERR,KARD(14),ACTION(14)	MISC 2
INTEGER ERFLAG,FERR,ACTION,PRTFIL	MISC 3
EQUIVALENCE (PRTFIL,FERR)	MISC 4
INTEGER A(84),B(84)	UNPAC 13

```

        ICHB = 1
        DO 10 I=1,84
          3(I) = 6H
10      CONTINUE
        DO 100 I=1,84
          N = I
          IF (A(I).EQ.1H3) RETURN
          IF (A(I).EQ.1H=) GO TO 70
          IF (A(I).EQ.1H ) GO TO 50
          IF (ICHB.GT.84) GO TO 220
          8(ICHB) = A(I)
50      CONTINUE
          ICHB = ICHB+1
          IF (ICHB.GT.84) RETURN
          GO TO 100
70      CONTINUE
          IF (ICHB.GT.71) GO TO 200
          ICHB=ICHB+1
90      ICHB = ((ICHB+9)/10)*10 + 1
100     CONTINUE
          GO TO 220
C
C          ERROR EXIT
C
200     CONTINUE
220     CONTINUE
        WRITE (FERR,310) A
310     FORMAT(28H UNPAC - CARD IMAGE TOO LONG/1X,84A1)
        ERFLAG = 1
        CALL TERM
        RETURN
      END
*ELT,1 DORMAN.UREAL
      SUBROUTINE UREAD (IUNIT,ICRD)
C
C          PROGRAMMER: VOIT

```

```

UNPAC 14
UNPAC 15
UNPAC 16
UNPAC 17
UNPAC 18
UNPAC 19
UNPAC 20
UNPAC 21
UNPAC 22
UNPAC 23
UNPAC 24
UNPAC 25
UNPAC 26
UNPAC 27
UNPAC 28
UNPAC 29
UNPAC 30
GWT25 4
UNPAC 31
UNPAC 32
UNPAC 33
UNPAC 34
UNPAC 35
UNPAC 36
UNPAC 37
UNPAC 38
UNPAC 39
UNPAC 40
UNPAC 41
UNPAC 42
UNPAC 43
UNPAC 44
UREAD 3
UREAD 5
UREAD 6
UREAD 7

```

C		UREAD	8
C		UREAD	9
	DIMENSION ICRD(14)	UREAD	10
	READ (IUNIT,10,END=200,ERR=200) (ICRD(I),I=1,14)	UREAD	17
	RETURN	UREAD	19
200	CONTINUE	UREAD	20
	DO 40 I=3,14	UREAD	21
	ICRD(I) = 6H	UREAD	22
40	CONTINUE	UREAD	23
	ICRD(1) = 6H\$END 0	UREAD	24
	ICRD(2) = 6HF FILE	UREAD	25
	RETURN	UREAD	26
10	FORMAT(13A6,A2)	UREAD	27
	END	UREAD	28
*ELT,I	DORMAN.USER	USER	3
	SUBROUTINE USER (IFLAG)	USER	5
C		USER	6
C	CONTROLS FILES	USER	7
C		USER	8
C	PROGRAMMER - S. WRAY	USER	9
C		USER	10
	COMMON /NAMES/ IVER,DNAME(2),MODNAM(2),BNAME(2)	NAMES	2
	INTEGER DNAME,MODNAM,BNAME	NAMES	3
	COMMON /FILES/ BASIC,MTAPE,FINAL,S1,S2	FILES	2
	INTEGER BASIC,MTAPE,FINAL,S1,S2	FILES	3
	COMMON /REST/TABLE,USES,FILE,END,DECK,DDECK,BLANK ,BATCH	REST	2
	INTEGER TABLE,USES,FILE,END,DECK,DDECK,BLANK,BATCH	REST	3
	COMMON /MISC/ERFLAG,FERR,KARD(14),ACTION(14)	MISC	2
	INTEGER ERFLAG,FERR,ACTION,PRTFIL	MISC	3
	EQUIVALENCE (PRTFIL,FERR)	MISC	4
	DIMENSION IU(3),IR(3)	USER	15
	DATA IU,IR/0,0,0,2,2,2/	USER	16
	IF(IFLAG.EQ.0) WRITE (PRTFIL,5)	USER	17
5	FORMAT(25H CREATE OPTION IS BLOCKED)	USER	18
	WRITE (PRTFIL,10)	USER	19
10	FORMAT(31H IF YOU WISH TO HAVE A DECK EXTRACTED FROM THE BANK/	USER	20

*49H ENTER THE NAME OF THE DECK, OTHERWISE ENTER DONE/9H READY -)	USER	21
CALL INCD (ACTION,5)	USER	22
IF(ACTION(1).EQ.4HDONE) RETURN	USER	23
IF((ACTION(1).EQ.MODNAM(1)).AND.(ACTION(2).EQ.MODNAM(2))) GO TO 15	USER	24
IF((ACTION(1).EQ.BNAME(1)).AND.(ACTION(2).EQ.BNAME(2))) RETURN	USER	25
FINAL = 14	USER	26
CALL USE (ACTION)	USER	27
RETURN	USER	28
15 M1 = 13	USER	29
FINAL = 14	USER	30
IU(1) = M1	USER	31
IU(2) = FINAL	USER	32
CALL INTBUF(IU,IR)	USER	33
REWIND M1	USER	34
CALL INCD(KARD,M1)	USER	35
IF((KARD(5).EQ.BNAME(1)).AND.(KARD(6).EQ.BNAME(2))) GO TO 20	USER	36
BNAME(1) = KARD(5)	USER	37
BNAME(2) = KARD(6)	USER	38
CALL USE (BNAME)	USER	39
20 CONTINUE	USER	40
M2 = 23	USER	41
IU(1) = M1	USER	42
IU(2) = M2	USER	43
IU(3) = FINAL	USER	44
CALL INTBUF(IU,IR)	USER	45
REWIND M1	USER	46
REWIND M2	USER	47
REWIND FINAL	USER	48
CALL EDITOK (FINAL,M1,M2)	USER	49
REWIND M2	USER	50
REWIND FINAL	USER	51
25 CALL INCD (KARD,M2)	USER	52
IF(ERFLAG.NE.0) RETURN	USER	53
CALL OUTCD (KARD,FINAL)	USER	54
IF(KARD(1).NE.END) GO TO 25	USER	55
BNAME(1) = ACTION(1)	USER	56

BNAME(2) = ACTION(2)	USER 57
RETURN	USER 58
END	USER 59
#ELT,I DORMAN.VALUE	VALUE 3
SUBROUTINE VALUE (A,V,IERR)	VALUE 5
C CONVERTS NUMERIC DATA FROM CODED TO FLOATING POINT FORMAT.	VALUE 6
C UPON ENTRY, A IS A 2-CELL ARRAY CONTAINING CODED VALUE IN (A6,A4)	VALUE 7
C FORMAT, 1-10 DIGITS LOCATED ANYWHERE IN 10-CHARACTER FIELD.	VALUE 8
C DECIMAL POINT OPTIONAL. NO EXPONENTS OR + OR - SIGNS. NO	VALUE 9
C BLANKS EMBEDDED BETWEEN DIGITS.	VALUE 10
C UPON EXIT, V CONTAINS CONVERTED VALUE IN FLOATING POINT. IERR IS A	VALUE 11
C ERROR FLAG--	VALUE 12
C IERR = 0 FOR NO ERROR.	VALUE 13
C IERR = -1 FOR COMPLETELY BLANK FIELD	VALUE 14
C IERR = 1,2,...,10 INDICATES POSITION OF ILLEGAL CHARACTER,	VALUE 15
C MULTIPLE DECIMAL POINTS, OR EMBEDDED BLANKS.	VALUE 16
C	VALUE 17
INTEGER A(1),C(19),CL(10),P(2)	VALUE 18
DATA C/19*1H /	VALUE 19
DATA CL/1H0,1H1,1H2,1H3,1H4,1H5,1H6,1H7,1H8,1H9/	VALUE 20
C	VALUE 21
C STORE CHARACTERS INDIVIDUALLY IN C(10)-C(19)	VALUE 22
C	VALUE 23
DECODE (6,70,A(1))(C(I),I=10,15)	VALUE 24
DECODE (4,70,A(2))(C(I),I=16,19)	VALUE 25
C	VALUE 26
C FIND POSITION OF FIRST AND LAST NON-BLANK CHARACTERS	VALUE 27
C	VALUE 28
NF=100	VALUE 29
NL=0	VALUE 30
DO 10 I=10,19	VALUE 31
IF (C(I).EQ.1H) GO TO 10	VALUE 32
NF=MIN0(I,NF)	VALUE 33
NL=I	VALUE 34
10 CONTINUE	VALUE 35
IF (NL.GT.0) GO TO 20	VALUE 36

	IERR=-1	VALUE 37
	GO TO 60	VALUE 38
C		VALUE 39
C	CHECK FOR ILLEGAL CHARACTERS, INCLUDING EMBEDDED BLANKS OR MULTIPLE	VALUE 40
C	DECIMAL POINTS	VALUE 41
C		VALUE 42
20	NDP=0	VALUE 43
	DO 50 I=NF,NL	VALUE 44
	DO 30 J=1,10	VALUE 45
	IF (C(I).EQ.CL(J)) GO TO 50	VALUE 46
30	CONTINUE	VALUE 47
	IF (C(I).NE.1H.) GO TO 40	VALUE 48
	NDP=NDP+1	VALUE 49
	IF (NDP.EQ.1) GO TO 50	VALUE 50
40	IERR=I-9	VALUE 51
	GO TO 60	VALUE 52
50	CONTINUE	VALUE 53
C		VALUE 54
C	REPACK CHARACTERS RIGHT-ADJUSTED IN FIELD OF 10 CHARACTERS.	VALUE 55
C		VALUE 56
	N1=NL-9	VALUE 57
	ENCODE (10,70,P) (C(I),I=N1,NL)	VALUE 58
	DECODE (10,80,P) V	VALUE 59
	IERR=0	VALUE 60
60	CONTINUE	VALUE 61
	RETURN	VALUE 62
70	FORMAT (10A1)	VALUE 63
80	FORMAT (F10.0)	VALUE 64
	END	VALUE 65
[ELT,I	DORMAN.ZEBRA	ZEBRA 3
#FOR,S	DORMAN.BLOCK,.BLOCK	ZEBRA 4
#FOR,S	DORMAN.DORMAN,.DORMAN	ZEBRA 5
#FOR,S	DORMAN.REPL,.REPL	ZEBRA 6
#FOR,S	DORMAN.USER,.USER	ZEBRA 7
#FOR,S	DORMAN.SAVER,.SAVER	ZEBRA 8
#FOR,S	DORMAN.ADDER,.ADDER	ZEBRA 9

#FOR,S DORMAN.DELET,.DELET	ZEBRA 10
#FOR,S DORMAN.EDITER,.EDITER	ZEBRA 11
#FOR,S DORMAN.CONV,.CONV	ZEBRA 12
#FOR,S DORMAN.OPT,.OPT	ZEBRA 13
#FOR,S DORMAN.INCRT,.INCRT	ZEBRA 14
#FOR,S DORMAN.ADDX,.ADDX	ZEBRA 15
#FOR,S DORMAN.TERM,.TERM	ZEBRA 16
#FOR,S DORMAN.LISTER,.LISTER	ZEBRA 17
#FOR,S DORMAN.A1TA6,.A1TA6	ZEBRA 18
#FOR,S DORMAN.A6TA1,.A6TA1	ZEBRA 19
#FOR,S DORMAN.CLOSE,.CLOSE	ZEBRA 20
#FOR,S DORMAN.ASSGN,.ASSGN	ZEBRA 21
#FOR,S DORMAN.DIFDEC,.DIFDEC	ZEBRA 22
#FOR,S DORMAN.FILBUF,.FILBUF	ZEBRA 23
#FOR,S DORMAN.FNDBUF,.FNDBUF	ZEBRA 24
#FOR,S DORMAN.INCD,.INCD	ZEBRA 25
#FOR,S DORMAN.INTBUF,.INTBUF	ZEBRA 26
#FOR,S DORMAN.LABLER,.LABLER	ZEBRA 27
#FOR,S DORMAN.LCRD,.LCRD	ZEBRA 28
#FOR,S DORMAN.OUTCD,.OUTCD	ZEBRA 29
#FOR,S DORMAN.PACCON,.PACCON	ZEBRA 30
#FOR,S DORMAN.RESTOR,.RESTOR	ZEBRA 31
#FOR,S DORMAN.SYNCBF,.SYNCBF	ZEBRA 32
#FOR,S DORMAN.UNPAC,.UNPAC	ZEBRA 33
#FOR,S DORMAN.UREAD,.UREAD	ZEBRA 34
#FOR,S DORMAN.GETGEN,.GETGEN	ZEBRA 35
#FOR,S DORMAN.LISTG,.LISTG	ZEBRA 36
#FOR,S DORMAN.LISTTC,.LISTTC	ZEBRA 37
#FOR,S DORMAN.USE,.USE	ZEBRA 38
#FOR,S DORMAN.ADD,.ADD	ZEBRA 39
#FOR,S DORMAN.REPLAC,.REPLAC	ZEBRA 40
#FOR,S DORMAN.DELETE,.DELETE	ZEBRA 41
#FOR,S DORMAN.COUNT,.COUNT	ZEBRA 42
#FOR,S DORMAN.EDITOK,.EDITOK	ZEBRA 43
#FOR,S DORMAN.VALUE,.VALUE	ZEBRA 44
#FOR,S DORMAN.SYNCD5,.SYNCD5	ZEBRA 45

#FOR,S DORMAN.SYNC1,.SYNC1	ZEBRA 46
#FOR,S DORMAN.DELCD,.DELCD	ZEBRA 47
#FOR,S DORMAN.INSERT,.INSERT	ZEBRA 48
#FOR,S DORMAN.CKSYN,.CKSYN	ZEBRA 49
#FOR,S DORMAN.EXTRAN,.EXTRAN	ZEBRA 50
#ELT,I DORMAN.USE	USE 3
SUBROUTINE USE(NAME)	USE 5
C PROGRAMMER - B. GOLD	USE 6
DIMENSION NAME(2),KG(20)	USE 7
DIMENSION IU(3),IR(3)	USE 8
INTEGER OUT	USE 9
COMMON /MISC/ERFLAG,FERR,KARD(14),ACTION(14)	MISC 2
INTEGER ERFLAG,FERR,ACTION,PRTFIL	MISC 3
EQUIVALENCE (PRTFIL,FERR)	MISC 4
COMMON /FILES/ BASIC,MTAPE,FINAL,S1,S2	FILES 2
INTEGER BASIC,MTAPE,FINAL,S1,S2	FILES 3
COMMON /REST/TABLE,USES,FILE,END,DECK,DDECK,BLANK ,BATCH	REST 2
INTEGER TABLE,USES,FILE,END,DECK,DDECK,BLANK,BATCH	REST 3
COMMON /NAMES/ IVER,DNAME(2),MODNAM(2),BNAME(2)	NAMES 2
INTEGER DNAME,MODNAM,BNAME	NAMES 3
INTEGER YES,NO	USE 14
INTEGER WORD	USE 15
INTEGER VERS	USE 16
DATA VERS /6HVERS10/	USE 17
DATA YES,NO/3HYES,2HNO/	USE 18
C	USE 19
C S1 AND S2 ARE SCRATCH TAPES	USE 20
C MTAPE IS THE TAPE ONTO WHICH THE MOD DECKS ARE COPIED	USE 21
C	USE 22
S1 = 25	USE 23
S2 = 26	USE 24
MTAPE = 27	USE 25
IU(1) = BASIC	USE 26
IU(2) = S1	USE 27
IU(3) = MTAPE	USE 28
IR(1) = 2	USE 29

```

      IR(2) = 2
      IR(3) = 2
      CALL INTBUF(10,IR)
      ERFLAG = 0
      REWIND BASIC
      REWIND MTAPE
      REWIND FINAL
      NG = 0
      LASTM = 0
      LASTB = 0
      NAME1 = NAME(1)
      NAME2 = NAME(2)
      IFLG = 1
      NTC = 0
C
C READ TABLE OF CONTENTS. STORE LIST OF ALL DECKS ON BASIC FILE IN
C ARRAY TC. STORE GENEALOGY OF DECK NAME IN ARRAY G.
C
      10 CALL INCD(KARD,BASIC)
         IF(ERFLAG.NE.0) RETURN
         IF(KARD(5).EQ.VERS) GO TO 10
         IF (KARD(1).EQ.END)      GO TO 20
         NTC = NTC+1
         IF (KARD(4).EQ.BLANK)  LASTB = NTC
         IF (KARD(4).EQ.USES)   LASTM = NTC
         IF ((KARD(2).NE.NAME1).OR.(KARD(3).NE.NAME2)) GO TO 10
         NG = NG+1
         KC(NG) = NTC
         NAME1 = KARD(5)
         NAME2 = KARD(6)
         GO TO 10
C
C READ MOD DECKS FROM BASIC FILE. IF REQUIRED FOR DECK NAME, STORE
C MOD DECK ON MTAPE.
C
      20 NDECK = 0

```

```

USE 30
USE 31
USE 32
USE 33
USE 34
USE 35
USE 36
USE 37
USE 38
USE 39
USE 40
USE 41
USE 42
USE 43
USE 44
USE 45
USE 46
USE 47
USE 48
USE 49
USE 50
USE 51
USE 52
USE 53
USE 54
USE 55
USE 56
USE 57
USE 58
USE 59
USE 60
USE 61
USE 62
USE 63
USE 64
USE 65

```

IF (NAME1.NE.BLANK) GO TO 200	USE 66
IFLG = 2	USE 67
IF (NG.EQ.1) GO TO 50	USE 68
IG = NG-1	USE 69
30 NDECK = NDECK+1	USE 70
IF (NDECK.GT.NTC) GO TO 220	USE 71
INDEX = LASTM - KG(IG) + 1	USE 72
WORD=NO	USE 73
IF (INDEX.EQ.NDECK) WORD=YES	USE 74
40 CALL INCD(KARD,BASIC)	USE 75
IF (ERFLAG.NE.0) RETURN	USE 76
IF (WORD.EQ.YES) CALL OUTCD(KARD,MTAPE)	USE 77
IF (KARD(1).NE.END) GO TO 40	USE 78
IF (WORD.EQ.NO) GO TO 30	USE 79
IG = IG-1	USE 80
IF (IG.GT.0) GO TO 30	USE 81
C	USE 82
C ALL MOD DECKS REQUIRED FOR DECK NAME HAVE BEEN PLACED ON MTAPE.	USE 83
C NOW FIND THE BASIC DECK REQUIRED.	USE 84
C	USE 85
50 IFLG = 3	USE 86
INDEX = LASTB - (KG(NG)-LASTM) + 1	USE 87
55 NDECK = NDECK+1	USE 88
IF (NDECK.GT.NTC) GO TO 220	USE 89
IF (NDECK.EQ.INDEX) GO TO 70	USE 90
60 CALL INCD(KARD,BASIC)	USE 91
IF (ERFLAG.NE.0) RETURN	USE 92
IF (KARD(1).NE.END) GO,55,60	USE 93
C	USE 94
C BASIC FILE IS NOW POSITIONED AT BEGINNING OF BASIC DECK REQUIRED.	USE 95
C COMBINED DECKS VIA EDIT.	USE 96
C	USE 97
70 IFLG = 4	USE 98
REWIND MTAPE	USE 99
REWIND FINAL	USE 100
BNAME(1) = NAME(1)	USE 101

BNAME(2) = NAME(2)	USE 102
IF (NG.GT.1) GO TO 90	USE 103
IU(2) = FINAL	USE 104
CALL INTBUF(IU,IR)	USE 105
80 CALL INCD(KARD,BASIC)	USE 106
IF(ERFLAG.NE.0) RETURN	USE 107
CALL OUTCU(KARD,FINAL)	USE 108
IF (KARD(1).EQ.END) RETURN	USE 109
GO TO 80	USE 110
90 IN = BASIC	USE 111
OUT = S1	USE 112
IG = NG	USE 113
100 IG = IG-1	USE 114
IF (IG.EQ.1) OUT = FINAL	USE 115
REWIND OUT	USE 116
IU(1) = IN	USE 117
IU(2) = MTAPE	USE 118
IU(3) = OUT	USE 119
CALL INTBUF(IU,IR)	USE 120
CALL EDITDK (IN,MTAPE,OUT)	USE 121
IF(ERFLAG.NE.0) RETURN	USE 122
IF (IG.EQ.1) RETURN	USE 123
IN = OUT	USE 124
OUT = S2	USE 125
IF (IN.EQ.S2) OUT = S1	USE 126
REWIND IN	USE 127
GO TO 100	USE 128
C	USE 129
C ERROR STOPS	USE 130
C	USE 131
200 WRITE (FERR,210) NAME1, NAME2, NAME	USE 132
210 FORMAT(28HCOULD NOT FIND DECK NAMED *2A6,20H* REQUIRED FOR DECK ,	USE 133
*2A6)	USE 134
ERFLAG = 1	USE 135
RETURN	USE 136
220 WRITE (FERR,230) NAME	USE 137


```
230 FORMAT(38H0INDEXING ERROR IN USE FOR DECK NAMED 2A6)  
    ERFLAG = 1  
    RETURN  
    END
```

```
USE 138  
USE 139  
USE 140  
USE 141
```